A Common Framework and Taxonomy for Multicriteria Scheduling Problems with Interfering and Competing Jobs: Multi-agent Scheduling Problems*

Paz Perez-Gonzalez† Jose M. Framinan

Industrial Management, School of Engineering,

University of Seville. Camino de los descubrimientos s/n. 41092 Seville, Spain

Abstract

Most classical scheduling research assumes that the objectives sought are common to all jobs to be scheduled. However, many real-life applications can be modeled by considering different sets of jobs, each one with its own objective(s), and an increasing number of papers addressing these problems has appeared over the last few years. Since so far the area lacks a unified view, the studied problems have received different names (such as interfering jobs, multi-agent scheduling, mixed-criteria, etc), some authors do not seem to be aware of important contributions in related problems, and solution procedures are often developed without taking into account existing ones. Therefore, the topic is in need of a common framework that allows for a systematic recollection of existing contributions, as well as a clear definition of the main research avenues. In this paper we review multicriteria scheduling problems involving two or more sets of jobs and propose an unified framework providing a common definition, name and notation for these problems. Moreover, we systematically review and classify the existing contributions in terms of the complexity of the problems and the proposed solution procedures, discuss the main advances, and point out future research lines in the topic.

Keywords: Scheduling; interfering jobs; multi-customer; multi-agent scheduling problems; sets of jobs; multicriteria.

*This is an Accepted Manuscript of an article published by Elsevier in European Journal of Operational Research Volume 235, Issue 1, 16 May 2014, Pages 1–16, available online: http://dx.doi.org/10.1016/j.ejor.2013.09.017

†Corresponding author. Tel. +3495 448 7212; fax: +3495 448 7329. E-mail address: pazperez@etsi.us.es

1 Introduction

The existence of several objectives is consubstantial to scheduling problems, as it can be seen from different definitions of the field, such as Pinedo (1995), where scheduling is defined as a decision-making process that has as a goal the optimization of one or more objectives. Therefore, it is not surprising that multicriteria scheduling problems have been widely studied in the scheduling literature (see e.g. the reviews by T'kindt and Billaut, 2001, 2002; Hoogeveen, 2005 and Minella et al., 2008). In all the problems analyzed in these surveys the criteria considered affect all jobs to be scheduled. Only Hoogeveen (2005) mentions the case of *multicriteria scheduling problems with two or more sets of jobs*. In these problems, two or more sets of jobs (not necessarily disjoint) have to be scheduled, each one with its own objective(s). Although this is a special case of multicriteria scheduling problems, the existence of several sets makes the problems rather different than their one-set counterpart as, in general, the complexity of these problems changes even if the objective functions are the same (Agnetis et al., 2004).

As discussed later in this paper, this type of scheduling problems arise in a number of real-life applications and therefore have been subject of interest by researchers and practitioners in the last few years. However, the lack of a unified framework has been a major deterrent for research advances in the field. There is not even a common name which has caused some contributions to ignore past works on the topic, as the keywords and title of the existing results make it difficult to conduct an extensive search (for instance, the same conclusion regarding a specific problem is independently shown by Agnetis et al., 2007a and Nong et al., 2011). Without a common definition and name, the notation and limits for this kind of problems are not clear, which may have hidden valuable contributions and makes the comparison among similar problems very difficult. In addition, this has caused that some scheduling problems dealing with two types of jobs, but with no interference among them, were considered part of the topic. For instance, in some problems described in Leung et al. (2010), the jobs in one set have their due date to be equal to their release date plus their processing times. Therefore, these jobs have to be processed in a specific (fixed) time interval and the remaining scheduling problem is how to schedule the jobs in the other set, which can be assimilated to a traditional scheduling problem with machine unavailability.

In this paper we will try to move towards an unified view on the topic that allows overcoming the above problems. More specifically, we 1) discuss the different definitions and approaches for the problem and provide a framework consisting of a single definition and notation, 2) give some complexity results and general properties, 3) review and classify the different contributions and results on the topic based on the aforementioned framework, and 4) point out the main research avenues in the field. By doing so, we expect to foster the research in this interesting and challenging scheduling area.

The remainder of the paper is structured as follows: Section 2 discusses the problems, their applications and the different names used in the literature. In Section 3 we present the notation and adapt the taxonomy presented by T'kindt and Billaut (2002) for multicriteria scheduling problems. The relationship among the complexities, single criteria and multicriteria scheduling problems is outlined in Section 4, where some general properties are presented as well. Literature is reviewed by classifying the problems into basic problems (discussed in Section 5) if there are no conditions for the machines or jobs, i.e. the most general case

indicating only the machine environment and the objectives considered for each set of jobs; and extended problems (discussed in Section 6), where specific conditions are imposed. Finally, Section 7 contains the conclusions and future research lines.

2 Fields of application and problem definition

We consider two or more set of jobs –not necessarily disjoint– competing or using common processing resources (machines). Each set of jobs has one objective, which may or may not be the same for each set. The objectives of some sets have to be optimized while others have to satisfy one or more constraints. This type of scheduling problems arise from many real-life fields of application (see Mor and Mosheiov, 2010):

- Supply chain scheduling: In a supply chain, a classical problem is to minimize overall manufacturing and distribution costs integrating production and delivery. If the customers are competing for a common processing resource then the problem implies interfering jobs. Fan (2010) presents a scheduling problem where the customers are placed at different locations such that delivery times are given. The objective is to minimize the sum of time between job's release and the delivery to the corresponding customer.

- Rescheduling: Rescheduling can be defined as the process of updating an existing production schedule in response to disruptions or other changes (Herrmann, 2006), such as the arrival of new jobs to be processed. Rescheduling usually implies more than one set of jobs, so a standard rescheduling problem can be formulated as a two-agent scheduling problem (Leung et al., 2010). In this problem there are usually two sets of jobs: existing jobs which have been already scheduled and new jobs to be scheduled. In this problem, two cases may be distinguished (Pinedo, 1995):

  - The starting times of the existing jobs cannot be modified ('frozen' jobs), so the problem can be considered as a single criteria problem subject to machine/job availability constraints (see e.g. Perez-Gonzalez and Framinan, 2009, 2010a; Perez-Gonzalez et al., 2011).
  - The starting times of the existing jobs can be modified (existing jobs can be rescheduled). If the same objective is considered both for existing and incoming jobs, the problem is again a scheduling problem with one set of jobs. However, in many situations it makes sense to employ different objectives for each set: Some performance measure is minimized for incoming jobs in order to obtain a short completion time for this set of jobs (e.g. minimizing their makespan or flowtime), while the objective for the existing jobs aims to minimize the disruption from their initial schedule. The usual way to achieve the latter objective is either to minimize their tardiness or impose that these jobs cannot be tardy (see e.g. Unal et al., 1997; Perez-Gonzalez and Framinan, 2010b); or to consider special disruption measures as the differences between the completion times of the old jobs in the sequence before rescheduling and the new sequence (such as in e.g. Hall and Potts, 2004; Yuan and Mu, 2007; Yuan et al., 2007; Mu and Gu, 2010).

- Telecommunications: Packet-switching networks usually support different applications, each one requiring the transmission of data packages that must reach their destination within some time limit.

The most important performance objective for some applications (such as file transfer or interprocess communication) is not to exceed certain mean delay, while for other applications (such as voice or video) is to achieve a specific loss rate. Therefore, the idea of several sets of jobs (packets belonging to applications) that must compete for the use of the same resource (the bandwidth) arises naturally. This problems have been addressed by Peha and Tobagi (1990); Peha (1995) and Meiners and Torng (2007). A similar problem is found in Arbib et al. (2004) for internet protocols, where one user wants to maximize the on-time packets transmitted to other user, while guaranteeing certain amounts of on-time packets to a third user.

- Maintenance scheduling: Some references in the literature address problems about scheduling jobs and preventive maintenance simultaneously (see e.g. Cassady and Kutanoglu, 2003; Ruiz et al., 2007), considering only one criteria for the jobs. However, since production and maintenance have common resources (the machines) and their activities are actually often conflicting, integrated production and maintenance cooperative scheduling is an example of interfering job problems when we consider a multiobjective approach. Khelifati and Bouzid-Sitayeb (2011a) simultaneously address the problem of scheduling production and preventive maintenance operations, taking into account both production and maintenance criteria. Since most machines have to be maintained at regular intervals (i.e. they require given periods of time on each machine), maintenance tasks can be modelled as *maintenance jobs* to be scheduled along with *production jobs*. Since maintenance tasks have to be performed within a time window, each maintenance job has both a release date and a due date (representing the earliest and latest time for the task, respectively). The natural objective for scheduling the set of maintenance jobs is thus to minimize a function of the deviation from their release and due dates, while production jobs are scheduled to minimize some performance measure. This approach is adopted by Wan et al. (2010). Kellerer and Strusevich (2010) give an interpretation of their specific interfering job problem where machine(s) is(are) subject to a compulsory maintenance during the planning period, the length and the deadline of the maintenance operations are given, and the Decision Maker has to decide when to start the maintenance period, while an objective related to the jobs has to be minimized.

It is to note that, there are several papers focusing on game theory aspects of the problems and its applications in industrial management, project scheduling, queuing setting, telecommunication services, economic markets, scheduling of trains, etc ... These papers are originally cited in Agnetis et al., 2000, 2004, 2007a,b), and they are subsequently cited often by authors dealing with interfering jobs scheduling problems, but it is to note that problems addressed there are not interfering jobs problems, at least in view of the definition given here. This is (another) side effect of the lack of a common framework for the topic.

Regarding the name employed to denote this type of problems, different authors have used different denominations:

- Peha (1995) call *heterogenous-criteria scheduling* to the problems where jobs are divided in two classes, and the performance measure differs from class to class. Earlier, Peha and Tobagi (1990) call *heterogeneous performance objectives* to the same problem, but without an explicit definition.

- The term *interfering job sets* first appeared in the review about multicriteria scheduling by Hoogeveen (2005), defined as a scheduling problem with two sets of jobs competing for processing on a single machine. *Interfering job sets* or *Interfering jobs* were later used by Balasubramanian et al. (2009); Elvikis et al. (2009, 2010b); Huynh-Tuong and Soukhal (2009a,b, 2010) in the multi-machine context. Their definitions are not homogeneous: In the first reference, the definition expresses that the jobs belong to disjoint classes or sets with a criterion associated with each set, while for the latter authors the sets are not disjoint, and there is an objective function for all jobs subject to minimizing another objective function for a subset of jobs, calling this case *global objective function* (Huynh-Tuong et al., 2011; Sadi et al., 2012).

- The most extended name is *two agent* or *multi agent* scheduling problems. In some cases it is called *competing agents* (Agnetis et al., 2004, 2009a; Fan, 2010; Huynh-Tuong et al., 2010; Kellerer and Strusevich, 2010; Mor and Mosheiov, 2010, 2011; Ding and Sun, 2011), being the main keyword *agent*. This concept was introduced by Agnetis et al. (2004) when addressing a scheduling problem in which two agents compete to perform their respective jobs on a common processing resource, each agent minimizing an objective function depending exclusively on the completion times of its jobs.

In addition, other –less employed– names have been suggested, such as 'competing users' (Shen, 1998; Agnetis et al., 2000), 'customers' (Baker and Smith, 2003), 'multiple job classes' (Soltani et al., 2010), 'multi-users' (Saule and Trystram, 2009), 'two families of jobs' (Yuan et al., 2005), 'mixed criteria scheduling problem' (Meiners and Torng, 2007), or even 'scheduling problem with a floating machine non-availability interval' (Kellerer and Strusevich, 2010), motivated by the fact that there is a machine maintenance interval which is not fixed. Finally, Arbib et al. (2004) do not give a name for the problem.

When trying to consolidate a name for the problem, it does not sound logic to choose one of these aforementioned less-employed names. Besides, multi-customer or multi-users do not imply a multicriteria problem, and 'class' or 'family' of jobs are terms used generally for batch scheduling which may be confusing. Similarly, 'heterogeneous criteria' and 'mixed criteria scheduling problem' imply different criteria for each set of jobs, so the case when the sets have the same objective is not considered.

While *agent* is the most extended name, we think that it could also be confusing, as there is an extensive literature on scheduling with multi-agent systems without relationship with our problem (see e.g. Coudert et al., 2002; Shen et al., 2006). Multi-agent system is a subfield of Distributed Artificial Intelligence where a network of individual agents share knowledge and communicate with each other to solve a problem beyond the scope of a single agent (Balaji and Srinivasan, 2010). This approach can be applied to scheduling in order to solve problems with competitive and negotiating algorithms controlled by agents (see e.g. Khelifati and Bouzid-Sitayeb, 2011a), but many references do not consider two or more set of jobs to be scheduled.

Finally, the term *sets* (or classes, or families) of jobs could be adopted, but it must be taken into account that there may be problems with several sets of jobs that do not necessarily compete for a resource, even if they have different objectives, and that these scheduling problem could be disaggregated into two single-criterion scheduling sub-problems.

Regardless if each job belongs to an agent, is a part of a customer order, an operation of a department or user, belongs to disjoint sets or not, etc. the key feature is that the job interferes with other jobs for the same resources. Although we think that *interfering jobs* is the most suitable term to name our problem, the most employed is *multi-agent scheduling problem* (MASP) so it is the term selected.

In the next section, we provide a common notation and taxonomy for these problems.

3 Notation and Taxonomy

Multi-agent Scheduling Problems (MASPs) consider $k = 2, \ldots, K$ sets of n^k jobs denoted $\mathcal{J}^1, \ldots, \mathcal{J}^K$. We define $n = \sum_{k=1}^{K} n^k$ and $\mathcal{J} = \mathcal{J}^1 \bigcup \cdots \bigcup \mathcal{J}^K$. Each set $\mathcal{J}^k$ has assigned a criterion f^k. If $\mathcal{J}^k = \mathcal{J}$ for some k, then f^k is denoted by f. In this case the problem is called interfering jobs problem with global objective function. Each job $j \in \mathcal{J}^k$ has to be processed on machine i with processing time p_{ij}^k, $i = 1, \ldots, m$. If there is only one machine, its processing time is denoted p_j^k. Additionally, we define $p^k = \sum_{i=1}^{m} \sum_{j \in \mathcal{J}^k} p_{ij}^k$ and $P = \sum_{i=1}^{m} \sum_{j \in \mathcal{J}} p_{ij}$. As many references deal with two disjoint sets of jobs ($k = 2$ $\mathcal{J}^1 \cap \mathcal{J}^2 = \emptyset$), we denote these sets as $\mathcal{J}^A$ and $\mathcal{J}^B$, each one with n^A and n^B jobs, and objective functions f^A and f^B, respectively.

Obviously, some special cases of this notation do not constitute MASPs. For instance, in the problem with two-sets, $\mathcal{J}^A = \emptyset$ (or $\mathcal{J}^B = \emptyset$) constitutes a single-criterion scheduling problem, while $\mathcal{J}^A = \mathcal{J}^B$ denotes a standard (i.e. not competing) multicriteria scheduling problem. Such cases will not be treated here.

A complete sequence $\sigma = [\sigma_1, \ldots, \sigma_n]$ is a permutation of all jobs in $\mathcal{J}$, where each σ_j, $j = 1, \ldots, n$, is in some $\mathcal{J}^k$. Given a sequence σ, the completion time of job σ_j in machine i is denoted as $C_{ij}(\sigma)$, being $C_j(\sigma)$ the completion time of job σ_j in the last machine on which this job needs to be processed. Without loss of generality, we omit σ in the completion times unless necessary.

A classification for MASPs can be obtained extending on the standard triplet $\alpha|\beta|\gamma$ notation by Graham et al. (1979) (see Pinedo, 1995 for a complete description). The α field represents the machine environment, i.e. including the single machine (1), identical machines in parallel (Pm), machines in parallel with different speeds (Qm), unrelated machines in parallel (Rm), flowshop (Fm), jobshop (Jm), or openshop (Om). We denote the case of the permutation flowshop (PFm) in order to obtain a clearer taxonomy of the problems with an empty β field (in the standard notation, the permutation condition in flowshops is indicated in the field β by the notation $prmu$).

β indicates conditions applying to jobs. If a condition is applied only in some sets, then it is indicated by a super-index. Usual restrictions are:

- Time-related conditions, such as release dates (r_j) and common due dates ($d_j = d$),
- Processing-times-related conditions: such as preemption ($prmp$); learning effect ($learning$) / deteriorating jobs ($deteriorating$) i.e. processing times decrease/increase with the starting time or the position of the job in σ; controlable processing times ($ctrl$), i.e. p_{ij}^k is a variable in the problem and can be chosen in a given interval $[\overline{p}_{ij}^k, \underline{p}_{ij}^k]$, with a cost of the compression of the processing times, c_{ij}^k, which is penalized in the objective function by $Ctrl_{sum}^k = \sum_{i=1}^{m} \sum_{j \in \mathcal{J}^k} c_{ij}^k (\overline{p}_{ij}^k - p_{ij}^k)$; and on-line problems

(*on-line*), when processing times only become known on arrival, opposite to off-line problems, which is the usual case, when processing times are known in advance.

- Batch-related constraints (*batch*). Parallel batch (*p-batch*) indicates that a machine can process up to b jobs simultaneously, being the processing time of a batch equal to the largest processing time of the jobs in the batch. In serial batching (*s-batch*), jobs are processed sequentially with a setup time for each batch (i.e. setup time between two jobs is equal to zero if both jobs are scheduled in the same batch). This case is also called family/part type-dependent setup times, including the case where a family is determined by a set of jobs. There may be batch capacity restrictions (denoted as $b \leq x$, otherwise we assume $b = \infty$), where each job has a size $size_j$. Finally, there may be incompatible sets (IS) if jobs from different sets cannot be placed in the same batch.

- Availability constraints. A case treated in the literature is that of forbidden intervals (I), denoting a time interval where no jobs can be scheduled. The number of forbidden intervals is denoted by n^*.

- Precedence. This condition can be applied to machines or jobs: Machine precedence is denoted as $m_i \mapsto m_j$ indicating that jobs must be processed by machine m_i before entering in machine m_j. Job precedence is denoted *prec*.

Usually, γ includes the objective function to be minimized. In our proposal, it contains a model indicating the performance measures f^k used for each set of jobs $\mathcal{J}^k$. Different problems or approaches can be defined using the functions f^k. In order to unify the notation, we integrate some of the multicriteria approaches in T'kindt and Billaut (2002) with those found in the MASP literature:

- **Linear Convex Combination (LCC) approach**: We denote $F_l(f^1, \ldots, f^K)$ if the objective is to minimize a linear convex combination of the K criteria, $\sum_{i=1}^{K} \lambda_i f^i$, where $\sum_{i=1}^{K} \lambda_i = 1$.

- **Epsilon-constraint (ϵ-constraint) approach**: We denote $\epsilon(f^1/f^2, \ldots, f^K)$, if the objective is to minimize f^1 subject to $f^k \leq \epsilon^k$, $\epsilon^k \geq 0$ for $k = 2, \ldots, K$. Constraints can be defined for all or for some f^k, i.e. $\epsilon^k = +\infty$ for some values of k. An instance of an ϵ-constraint problem may not have feasible solutions, and it is denoted feasible if there is at least one feasible solution.

- **Pareto approach:** We denote $\#(f^1, \ldots, f^K)$ if the objective is to enumerate all Pareto optima. A schedule σ is weak *Pareto optimum* if and only if $\nexists \sigma'$ such that $\forall k = 1, \ldots, K$, $f^k(\sigma') < f^k(\sigma)$, and it is strict *Pareto optimum* (or *efficient solution* or *nondominated solution*) if and only if $\nexists \sigma'$ such that $\forall k = 1, \ldots, K$, $f^k(\sigma') \leq f^k(\sigma)$ ($f^k(\sigma') < f^k(\sigma)$), with at least one strict inequality.

Other approaches found in the literature are **Goal Programming** and **Lexicographical approach**. The goal programming approach, denoted $GP(f^1, \ldots, f^K)$, also named feasibility model or decision model, is the special case $\epsilon(f/f^1, \ldots, f^K)$, with f constant. The goal here is to find a feasible schedule σ satisfying $f^k \leq \epsilon^k$, $k = 1, \ldots, K$. The lexicographical approach, denoted $Lex(f^1, \ldots, f^K)$, minimizes all criteria in the given order.

Note that the order in which f^k are considered is only relevant for the ϵ-constraint and the lexicographical approaches, as different orders imply different problems.

For each approach, different forms of f^k can be considered. Cheng et al. (2006) distinguish two types depending on the functions g_j^k applied to the completion times C_j^k, $j \in \mathcal{J}^k$, $k = 1, \ldots, K$, i.e.: *max-form*: $\max f^k = \max_{1 \leq j \leq n^k} g_j^k(C_j^k)$, and *sum-form*: $\sum f^k = \sum_{1 \leq j \leq n^k} g_j^k(C_j^k)$. When the jobs are weighted, we denote the functions $\max w^k f^k$ and $\sum w^k f^k$ respectively.

If g_j^k are non-decreasing functions of C_j^k for all $j \in \mathcal{J}^k$, then they are called **regular performance measures** (see e.g. Pinedo, 1995). We denote a regular performance measure with capital letter.

Among non-regular performances measures, only those related to *earliness*, defined as $E_j = \max\{0, d_j - C_j\}$, have been found in the review. These performance measures, jointly with the related to the tardiness defined below, are called *just in time* performance measures (see e.g. T'kindt and Billaut, 2002 for more information).

Table 1 summarizes the regular and non-regular performance measures encountered in the literature. The usual definitions for *lateness* ($L_j = C_j - d_j$), *tardiness* ($T_j = \max\{C_j - d_j, 0\}$), and *unit penalty* ($U_j$ yields 1 if $C_j > d_j$, 0 otherwise) apply.

Type	**Function form**		
Regular		**Due date independent**	
	Max-form	*Maximum cost*	$\max F_j^k = \max_{j \in \mathcal{J}^k} g_j^k(C_j)$	
		Makespan (Weighted)	$\max C_j^k = \max_{j \in \mathcal{J}^k} C_j$	$(\max w_j^k C_j^k = \max_{j \in \mathcal{J}^k} w_j C_j)$
	Sum-form	*Sum of costs*	$\sum F_j^k = \sum_{j \in \mathcal{J}^k} g_j^k(C_j)$	
		Flowtime (Weighted)	$\sum C_j^k = \sum_{j \in \mathcal{J}^k} C_j$	$(\sum w_j^k C_j^k = \sum_{j \in \mathcal{J}^k} w_j C_j)$
		Due date dependent		
	Max-form	*Maximum lateness*	$\max L_j^k = \max_{j \in \mathcal{J}^k} L_j$	
		Maximum tardiness	$\max T_j^k = \max_{j \in \mathcal{J}^k} T_j$	
	Sum-form	*Total tardiness* (Weighted)	$\sum T_j^k = \sum_{j \in \mathcal{J}^k} T_j$	$(\sum w_j^k T_j^k = \sum_{j \in \mathcal{J}^k} w_j T_j)$
		Number of tardy jobs (Weighted)	$\sum U_j^k = \sum_{j \in \mathcal{J}^k} U_j$	$(\sum w_j^k U_j^k = \sum_{j \in \mathcal{J}^k} w_j U_j)$
Non regular		**Due date dependent**		
	Max-form	*Maximum earliness*	$\max E_j^k = \max_{j \in \mathcal{J}^k} E_j$	
	Sum-form	*Total earliness* (Weighted)	$\sum E_j^k = \sum_{j \in \mathcal{J}^k} E_j$	$(\sum E_j^k = \sum_{j \in \mathcal{J}^k} w_j E_j)$

Table 1: Usual performance measures for MAPSs

Finally, we also analyse performance measures employed in the rescheduling literature in order to include them in the notation proposed. In rescheduling, usually two sets of jobs are considered: old jobs $\mathcal{J}^O$, and new jobs $\mathcal{J}^N$, as well as two schedules: an initial sub-sequence σ_o, formed by jobs in $\mathcal{J}^O$, and a final sequence obtained when jobs have been rescheduled σ_n, formed by jobs in $\mathcal{J} = \mathcal{J}^O \bigcup \mathcal{J}^N$. There are two approaches depending on σ_o (Hall and Potts, 2004): to consider a *single disruption*, so the original schedule is optimal with respect to the initial scheduling objective, or to consider the most recent of *multiple disruptions*, so the initial schedule is regarded as arbitrary due to previous disruptions.

Different ways to measure the disruption of jobs in $\mathcal{J}^O$ can be considered:

1. No tardy jobs constraint. In this case, jobs in $\mathcal{J}^O$ cannot be late, which means that $C_j \leq d_j, \forall j \in \mathcal{J}^O$. There are several (equivalent) ways to model this constraint, such as $\max L_j^O \leq 0$, $\max T_j^O = 0$ (equivalent to $\max T_j^O \leq 0$), $\sum T_j^O = 0$ (equivalent to $\sum T_j^O \leq 0$), and $\sum U_j^O = 0$ (equivalent to $\sum U_j^O \leq 0$). All of these models have a weighted counterpart. Note that these constraints are on the form of an ϵ-constraint approach. If the objective is to minimize f subject to no tardy jobs for the set $\mathcal{J}^O$, it is denoted as $0(f/f^O)$, with $f^O \in \{\max L_j, \max T_j, \sum T_j, \sum U_j, \max w_j T_j, \sum w_j T_j, \sum w_j U_j\}$.

2. Sequence disruption: $D_j(\sigma_o, \sigma_n) = |y - x|$ with y the position of the job j in σ_o, and x its position in σ_n. When there is no ambiguity, the notation is simplified to D_j, with objectives $\max D_j$ and $\sum D_j$.

3. Time disruption: $\Delta_j(\sigma_o, \sigma_n) = |C_j(\sigma_n) - C_j(\sigma_o)|$. When there is no ambiguity, the notation is simplified to Δ_j, with objectives $\max \Delta_j$ and $\sum \Delta_j$.

Depending on the approach, machine setting, objective function, etc, there are many different problems in the literature. A broad classification is to divide the problems into basic problems (i.e. when there are no conditions for the machines or jobs, i.e. the most general case indicating only the machine environment and the objectives considered for each set of jobs) and extended problems (i.e. where specific conditions are imposed). Prior to this classification, it is useful to recall the complexity of the different problems and to derive general properties that may apply to some approaches. Both aspects are discussed in the next section.

4 Complexity and General Properties

4.1 Complexity

Many contributions in the literature about MASPs are devoted to classify and/or determine the complexity of different problems. In the same way as in single criterion and classical multicriteria scheduling problems, the complexities of some MASPs can be determined by reductions to known problems. In this subsection we intend to give a first step to determine the relationship between the complexities of MASPs and known problems.

Complexity theory is based on languages theory and Turing machines (see Garey and Johnson, 1979). A problem belongs to a class of complexity, which informs us of the complexity of the "best algorithm" able to solve it (T'kindt and Billaut, 2002). We know the complexity of a problem if it can be reduced from/to another problem with a known complexity. A search (decision) problem π' is NP-hard (NP-complete) if and only if another search (decision) problem π is NP-hard (NP-complete) and a reduction of π towards π' exists. The problem π' is at least as difficult to solve as the problem π. NP-hard problems are also called NP-hard in the ordinary sense, ordinary NP-hard, binary NP-hard problems, or weakly NP-hard. A subclass of NP-hard problems are the so-called strongly NP-hard problems (NP-hard in the strong sense or unary NP-hard). For detailed definitions we refer the reader to Garey and Johnson (1979); Gawiejnowicz (2008); T'kindt and Billaut (2002).

In order to establish the relationship among the complexities of single-criterion, multicriteria and MASPs, we use the following notation:

- D_i: Decision problem associated to the criterion f^i.
- SC_i: Single criterion problem with criterion f^i.
- MC_*: Multicriteria scheduling problem (i.e. only one set of jobs is considered), * denoting the approach (see section 3).
- $MASP_*$: MASP (i.e. two or more sets of jobs are considered), * denoting the approach.
- $MASP_*^{DJ}$: MASP with disjoint sets of jobs, * denoting the approach.
- $MASP_*^{GO}$: MASP with global objective, * denoting the approach.

4.1.1 Relation to single-criterion scheduling problems

The special case of an MASP when all sets of jobs except one are empty is a single-criterion scheduling problem. T'kindt and Billaut (2002) (pp. 39-42) give a complete study about the complexities of single-criterion scheduling problems, providing the reduction trees depending on the machine setting, on the constraints and criteria. Since reduction tree of machine settings and reduction tree of constraints can be applied to multicriteria problems as well as to MASPs, they are not discussed here. Note that the latter allow us to identify the relationship among the complexities of basic and some simple cases of extended problems (under the same machine and criteria conditions). However, application of the reduction tree of criteria to multicriteria and therefore, to MASPs is not direct, since it depends on the approach.

Within each approach, reductions can be developed whenever we consider two problems in which all sets (except one) involve the same objective functions, i.e. if $\alpha|\beta|f$ is reduced from $\alpha|\beta|g$, then problem $\alpha|\beta|*(f^1,\ldots,f^k,\ldots,f^K)$ is reduced from $\alpha|\beta|*(f^1,\ldots,g^k,\ldots,f^K)$. Note that if all K sets in each problem have the same function, then $\alpha|\beta|*(f^1,\ldots,f^k,\ldots,f^K)$ is reduced from $\alpha|\beta|*(g^1,\ldots,g^k,\ldots,g^K)$.

4.1.2 Relation to multicriteria scheduling problems

Another special case of MASPs is when all sets of jobs are equal. Then, the problem is equivalent to a standard multicriteria scheduling problem. However, there are more MASPs than multicriteria scheduling problems, since it does not make sense to consider multicriteria problems for the ϵ-constraint and LCC approaches with equal objectives, while the multi-agent counterpart is well-defined.

The relationship among the complexities of the different approaches is the same for multicriteria scheduling problems than for MASPs. We have adapted the reduction tree of multicriteria problems given by T'kindt and Billaut (2002) to MASPs (see Figure 1, where a solid arrow from problem A to problem B indicates that a reduction exists from A to B). As it is possible to associate a decision problem with an optimisation problem, the relation among their complexities is represented in Figure 1 by dotted arrows. Therefore, the complexity of most problems concerning the goal programming approach are a straightforward consequence of the corresponding ϵ-constraint version.

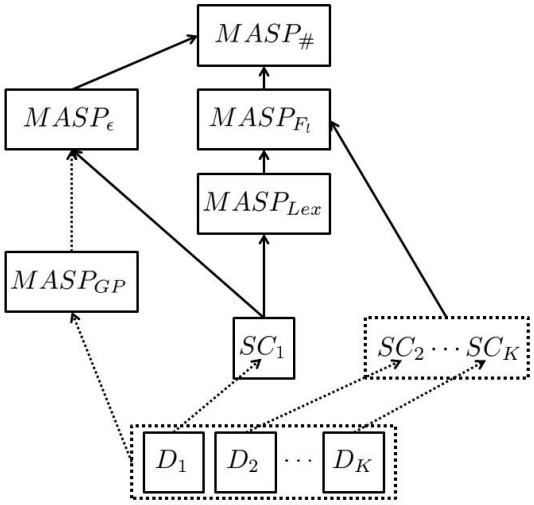

Figure 1: Reduction tree for MASPs, adapted from T'kindt and Billaut (2002)

Although the reduction tree for multicriteria and MASPs is the same, in general, the complexities of the MASPs are not the same as that of the corresponding multicriteria problems even if the objective functions are the same (Agnetis et al., 2004). Agnetis et al. (2004) give some examples where the complexities are not the same in the case of basic problems of the ϵ-constraint approach. Therefore, the tables with the complexities for bicriteria basic problems for LCC, Lexicographical and ϵ-constraint approaches provided by T'kindt and Billaut (2002) (pp. 116, 117) do not apply for MASPs. However, for the special case of MASPs with global objective, Huynh-Tuong et al. (2011) develop some reductions to these problems from single-criterion, bicriteria and MASPs with two disjoint sets of jobs. Only ϵ-constraint and LCC approaches have been considered. The reductions have been adapted for general cases with more than two objective functions in Figure 2, where solid arrow applies for regular measures, and dashed arrow implies reductions for objective functions belonging to the set $\{\max L_j, \sum w_j C_j, \sum w_j T_j, \sum U_j, \sum w_j U_j\}$.

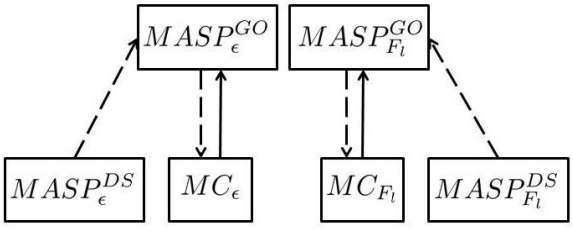

Figure 2: Simple reductions from multicriteria and MASPs with disjoint sets to MASP with global objective, adapted from Huynh-Tuong et al. (2011)

4.1.3 Solution Procedures

An algorithm is a finite procedure that finds a solution to an arbitrary instance of a problem. In the literature, there are different methods to solve scheduling problems. Gawiejnowicz (2008), for example, presents a classification of different types of algorithms applied to scheduling in the literature (in particular to time-dependent scheduling problems), which can be extended to scheduling problem in general, and to MASPs in particular: Exact algorithms find the exact solution of a problem (not only feasible solutions). Polynomial-time algorithms are examples of exact algorithms that find the exact solution in polynomial-time. For the problems for which no polynomial-time algorithms are known, exact solutions can be found by enumerative, branch-and-bound (B&B) or dynamic programming algorithms. Pseudopoynomial-time algorithms are applied to NP-hard problems, and, in general, a problem which is strongly NP-hard cannot be solved by them. Approximation algorithms are polynomial algorithms that generate an approximate solution that is close to an optimal solution, measured by the worst-case ratio (approximation ratio), which is a number in single criterion problems. In multicriteria and MASPs with Pareto and Goal Programming approaches and objectives $(f_1, \ldots, f_K)$, the $(\beta_1, \ldots, \beta_K)$-approximation schedule is the schedule which provides an approximation ratio of β_k for objective f_k (see e.g. Lee et al. (2009); Saule and Trystram (2009)). A family of approximation algorithms which generate solutions as close to the optimal one as desired is called an approximation scheme. There exist two types of approximation schemata: polynomial-time approximation scheme (PTAS) and fully polynomial-time approximation scheme (FPTAS). Finally, Heuristics and Metaheuristics are algorithms with unknown worst-case ratio. It is not possible to predict the behavior of this algorithm for all instances and their efficiency is evaluated by computational experiments. Some examples found in the MASP literature are Genetic Algorithms (GA), Simulated Annealing (SA), Tabu Search (TS), Variable Neighborhood Search (VNS), or Ant Colony Optimization (ACO).

All previous methods can be applied to MASPs regardless the approach. However, the Pareto approach is the most difficult (see Figure 1), and it is not easy to find methods to determine the set of Pareto optima for MASPs. Usually, a straightforward way is to solve repeatedly instances of the corresponding ϵ-constraint problem for decreasing values of ϵ (for example, see Elvikis et al., 2010a, 2009; Agnetis et al., 2004). Theoretically, an optimal solution to the ϵ-constraint problem is a weak Pareto optimum. However, it is possible to assign an ϵ value to each strict Pareto optimum that it is an optimal solution to the associated ϵ-constraint problem (Ehrgott, 2005). For example, if we consider an MASP with two disjoint sets of jobs on a single machine, the problem of finding one schedule which is also non dominated among the optimal schedules of the ϵ-constraint version can be always addressed by binary search (Agnetis et al., 2004).

In some cases, only the cardinality (size) of the set of Pareto optima is provided. This size depends on the specific problem, and it can be polynomially bounded or not. Note that, when the number of tardy jobs is considered at least for one set of jobs, it is clear that the number of non-dominated schedules is linear.

Agnetis et al. (2004) give a scheme for enumerating the set of Pareto optima when the problem involves two sets of jobs on a single machine. Figure 3 presents a generalization of this scheme for K sets of jobs, $1||\epsilon(f^1/f^2, \ldots, f^K)$, considering $\epsilon = (\epsilon_2, \ldots, \epsilon_K)$ the bounds of $f^2, \ldots, f^K$, and a given vector $\delta = (\delta_2, \ldots, \delta_K)$ with $\delta_k > 0,\ \forall k = 2, \ldots, K$. This method depends on the corresponding ϵ-constraint problem, and when it is

polynomially solvable, the scheme turns out to be polynomial for those problems with a polynomial number of Pareto optima. Then, this method provides a way to determine the cardinality of the set of Pareto optima in some cases.

```
{
S := ∅;  Q := +∞;  i := 0
while 1||ϵ(f^1/f^2, ..., f^K) is feasible
    {
    i := i + 1
    σ^i := strict pareto solution to 1||ϵ(f^1/f^2, ..., f^K)
    S := S ∪ σ^i
    for k = 2 to K
        ϵ'_k := f^k(σ^i)
    ϵ := ϵ' − δ
    }
}
```

Figure 3: Scheme for enumerating strict pareto optima for the problem $1||\#(f^1, f^2, \ldots, f^K)$, adapted from Agnetis et al. (2004)

4.2 General Properties

For certain types of MASPs, some properties on the structure of their optimal solutions can be proved. Some of these properties have been proved by Baker and Smith (2003) for **single machine** MASPs. Baker and Smith (2003) consider only the LCC approach and some regular performance measures. Here we provide a generalisation of these properties for all approaches and regular performance measures. Standard proofs by contradiction apply. These properties may be a starting point to study more complex problems (different layouts or extended problems) to determine their complexities and even to be applied to solution methods (exact or approximate) to solve these problems.

The starting point for obtaining these properties is the problem $1||\max C_j$, since any job sequence provides the same makespan, and therefore, all schedules are optimal. Consequently, it can be observed that this objective does not have influence on the solution of multicriteria problems and MASPs with two non disjoint sets and makespan for the global objective function. Then, these problems have the same complexity as the single criterion problem for the other objective. For the rest of the cases, the complexity of MASPs considering makespan for some set $\mathcal{J}^k$ depends on the approach selected. The following properties are related to the sequences of jobs in optimal solutions according to the regular performance measures selected:

Property 4.1 *There is an optimal solution for the MASP considering* $\max C_j$ *for any set* $\mathcal{J}^k$*, in which the jobs belonging to this set are processed consecutively.*

The previous property implies that, the structure of an optimal solution to a problem with two disjoint sets of jobs considering makespan for $\mathcal{J}^B$, is $\mathcal{J}^A_{prec} \cup \mathcal{J}^B \cup \mathcal{J}^A_{succ}$, where $\mathcal{J}^A_{prec} \cup \mathcal{J}^A_{succ} = \mathcal{J}^A$ and $\mathcal{J}^A_{prec} \cap \mathcal{J}^A_{succ} = \emptyset$. Using this property, we can conclude that the optimal solution to these problems is completely defined by

the partition $(\mathcal{J}^A_{prec}, \mathcal{J}^A_{succ})$ of the set $\mathcal{J}^A$, a result provided by Agnetis et al. (2009a) for the specific case $1||\epsilon(\max L^A_j / \max C^B_j)$.

Based on this property, it can be observed that the complexity of any MASP with LCC approach for two disjoint sets of jobs and makespan as one of the criteria is the same as the corresponding single criterion problem for the second criterion (Baker and Smith, 2003). This result can be generalized as follows: the complexity of any MASP with LCC approach for more than two disjoint sets of jobs and makespan criterion for $\mathcal{J}^k$ is the same as the corresponding problem without the k-th set (Cheng et al., 2008).

However, for the ϵ-constraint approach (and consequently for the Pareto approach) the problems cannot be reduced from the single criterion case even for two sets of jobs. For example, $1||\epsilon(\sum w_j C^A_j / \max C^B_j)$ (see Section 5) is NP-hard (Agnetis et al., 2004), and $1||\#(\sum w_j C^A_j, \max C^B_j)$ has an exponential number of non-dominated solutions (Agnetis et al., 2007b), while $1||\sum w_j C_j$ is polynomial (Lawler et al., 1993).

Property 4.2 *There is an optimal solution for the MASP considering* $\max L_j$ *for one of the sets,* $\mathcal{J}^k$*, in which the jobs belonging to this set are ordered by non-decreasing values of the due dates* d^k_j*, that is, EDD order (even though they may not appear consecutively in the schedule).*

Property 4.3 *There is an optimal solution for the MASP considering* $\sum C_j$ *for one of the sets,* $\mathcal{J}^k$*, in which the jobs belonging to this set are ordered by non-decreasing values of their processing times* p^k_j*, that is, SPT order (even though they may not appear consecutively in the schedule).*

Property 4.3 cannot be adapted for $\sum w_j C_j$ using the WSPT order for any set of processing times and weights (for a counterexample see Agnetis et al., 2007a). Baker and Smith (2003) tried to prove it for $1||F_l(\sum w_j C^A_j, F^B)$ when $F^B \in \{\max C_j, \max L_j, \sum w_j C_j\}$ if $p^A_i \leq p^A_j$ and $\frac{p^A_i}{w^A_i} \leq \frac{p^A_j}{w^A_j}$. However, Yuan et al. (2005) give a counterexample showing the property to be incorrect. They provide a weakened version of the property, and their proof can be generalized to other approaches when all functions are regular:

Property 4.4 *There is an optimal solution for the MASP considering* $\sum w_j C_j$ *for one of the sets,* $\mathcal{J}^k$*, with* $p^k_i \leq p^k_j$ *whenever* $w^k_i > w^k_j$*, in which the jobs belonging to this set are ordered by non-increasing values of* p^k_j*, that is, in SPT order.*

5 Basic problems

In this Section we review *basic problems* where no conditions on machines or jobs are imposed (i.e. the field β is empty) and the problems have the form $\alpha||\gamma$. We classify the problems in disjoint and non-disjoint sets of jobs. The former case can be known as competing sets of jobs, and the latter as interfering sets of jobs.

For each case, we classify the problems according to the approach: LCC, ϵ-constraints and Pareto. Some results about the GP approach are included whenever appropriate.

$f^A \backslash f^B$	$\max L_j$	$\max T_j$	$\sum C_j$	$\sum w_j C_j$	$\sum U_j$	$\sum w_j U_j$
$\max C_j$	$\mathcal{O}(n^B \log n^B)$	$\mathcal{O}((n^A)^3 n^B + n^A (n^B)^3)$	$\mathcal{O}(n^3 \log n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n^3 \log n)$	NP
$\max L_j$	$\mathcal{O}(nn^A n^B)$	$\mathcal{O}((n^A)^3 n^B + n^A (n^B)^3)$	$\mathcal{O}(nn^A n^B)$	NP	$\mathcal{O}(n^3 \log n)$	NP
$\max T_j$		$\mathcal{O}((n^A)^3 n^B + n^A (n^B)^3)$	$\mathcal{O}(n^3 \log n)$	Open	$\mathcal{O}(n^3 \log n)$	NP
$\sum C_j$			$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	Open	NP
$\max w_j C_j$				$\mathcal{O}(n \log n)$	Open	NP
$\sum U_j$					$\mathcal{O}(n^4)$	NP
$\sum w_j U_j$						NP

Table 2: Complexity for basic MASPs with two disjoint set of jobs on a single machine for LCC approach

5.1 Disjoint (Competing) sets of jobs

5.1.1 LCC approach

In most of the problems addressed in this approach, the layout is the one-machine case. Complexities for two sets of jobs are summarised in Table 2 where f^A and f^B are regular functions. As the problem is symmetric, we have indicated the complexities in the upper part of a triangular matrix. These problems have been studied by Baker and Smith (2003); Yuan et al. (2005); Agnetis et al. (2007a) and Soltani et al. (2010).

Apart from the cases in Table 2, Agnetis et al. (2007a) show that the general case $1||F_l(\max F_j^A, \max F_j^B)$ (with $\max F_j^A$ and $\max F_j^B$ regular functions) can be solved in $\mathcal{O}(n^3 \log n)$. Additionally, for more than two sets of jobs, Cheng et al. (2008) show that the complexity of $1||F_l(\max F_j^1, \ldots, \max F_j^K)$ is $\mathcal{O}((n_1 n_2 \ldots n_K)^K \log n)$ for $F_j^k \in \{T_j^k, L_j^k\}$, $\forall k = 1, \ldots, K$.

As discussed in Section 4, $1||F_l(\max C_j^A, \max C_j^B)$ is trivial (Baker and Smith, 2003). Moreover, this problem for K sets of jobs ($K > 2$) is polynomially solvable in $\mathcal{O}(n + K \log K)$ (Cheng et al., 2008). However, the weighted version $1||F_l(\max w_j^1 C_j^1, \ldots, \max w_j^K C_j^K)$ is proved to be strongly NP-hard by Cheng et al. (2008) (since $1||GP(\sum w_j^A C_j^A, \max L_j^B)$ is strongly NP-complete). The cases $1||F_l(\max F_j^A, \sum U_j^B)$ and $1||F_l(\max F_j^A, \sum C_j^B)$ are shown to be both polynomially solvable in $\mathcal{O}(n^3 \log n)$ by Agnetis et al. (2007a). Although $1||F_l(\max C_j^A, \sum w_j^B C_j^B)$ is polynomially solvable by the WSPT rule (Baker and Smith, 2003), the problem $1||F_l(\max w_j^A C_j^A, \sum w_j^B C_j^B)$ is proved to be strongly NP-hard by Feng and Yuan (2007). Nong et al. (2011) give a 2-approximation algorithm for this problem. However, they prove that this problem is NP-hard in the ordinary sense when n^A is fixed, and develop a PTAS for the case where preemption of jobs in $\mathcal{J}^B$ is allowed, which can be applied to the case without preemption.

Soltani et al. (2010) develop a GA and SA for $1||F_l(\max L_j^A, \sum w_j^B C_j^B)$, which can be seen in Table 2 that it is NP-hard. Baker and Smith (2003) reduce $1||F_l(\max C_j^1, \max L_j^2, \sum w_j^2 C_j^3)$ to the latter problem by consolidating jobs in $\mathcal{J}^1$ as a single job j^A with processing time $p^1 = \sum_{j \in \mathcal{J}^1} p_j$ and weight $w^1 = 1$ and considering $\mathcal{J}^A = \mathcal{J}^2$ and $\mathcal{J}^B = j^A \bigcup \mathcal{J}^3$.

Regarding other layouts, only the problem $PF2||F_l(\max C_j^A, \max C_j^B)$ has been studied by Luo et al. (2011), showing that it is NP-hard. They provide a dynamic programming algorithm with running time $\mathcal{O}(nP^4)$, $P = \sum_{j \in \mathcal{J}} p_j$. They also develop a FPTAS with running time $\mathcal{O}(n^5/\delta^8)$, being $\delta > 0$ the worst-case

$f^A \backslash f^B$	$\max F_j$	$\sum C_j$	$\sum w_j C_j$	$\sum T_j$	$\sum U_j$	$\sum w_j U_j$
$\max F_j$	$\mathcal{O}((n^A)^2 + n^B \log n^B)$	$\mathcal{O}(n^A \log n^A + n^B \log n^B)$	NP	NP	$\mathcal{O}(n^A \log n^A + n^B \log n^B)$	NP
$\sum C_j$		NP	NP	NP	NP	NP
$\sum w_j C_j$			NP	NP	s-NP	s-NP
$\sum T_j$				NP	NP	NP
$\sum U_j$					$\mathcal{O}(nn^A n^B)$	NP
$\sum w_j U_j$						NP

Table 3: Complexity for basic MASPs with two disjoint set of jobs on a single machine for ϵ-constraint approach

ratio.

5.1.2 ϵ-constraint approach

In this section, we review the problem when the sets of jobs are disjoint considering the ϵ-constraint approach. Most studies are devoted to one-machine layout and two sets of jobs. Table 3 summarizes the complexities for $1||\epsilon(f^A/f^B)$ when f^A and f^B are regular functions. For the single machine case, Agnetis et al. (2004) proved that the symmetric problems are equivalent, since it is possible to reduce $1||\epsilon(f^B/f^A)$ from $1||\epsilon(f^A/f^B)$ by a binary search in a logarithmic number of instances. Then, in the same way that the LCC approach, we have indicated the complexities in the upper part of a triangular matrix. These problems have been studied by Agnetis et al. (2004); Ng et al. (2006); Agnetis et al. (2007a, 2009a); Leung et al. (2010).

The general case $1||\epsilon(\max F_j^A/\max F_j^B)$, with $\max F_j$ regular is polynomial (Agnetis et al., 2004, 2007a). This result is extended by Agnetis et al. (2004) for more than two sets of jobs, giving Agnetis et al. (2007b) an algorithm for the goal programming counterpart. Moreover, Agnetis et al. (2004) prove that $1||\epsilon(\sum w_j^A C_j^A/\max F_j^B)$ is NP-hard by proving that the goal programming counterpart is NP-complete, although $1||GP(\sum C_j^1, \max F_j^2, \ldots, \max F_j^K)$ is polynomial in $\mathcal{O}(n^1 \log n^1 + n' \log n')$, with $n' = \sum_{k=2}^{K} n^k$ (Agnetis et al., 2007b).

Although $1||\epsilon(\sum w_j^A C_j^A/\max F_j^B)$ is NP-hard, the case with $\max L_j^B$ is shown to be strongly NP-hard by Ng et al. (2006). The case $1||\epsilon(\sum w_j^A C_j^A/\max C_j^B)$ is addressed by Kellerer and Strusevich (2010) by means of an FPTAS. They show it to be equivalent to a single criterion problem with one fixed non-availability machine interval in the case that the non-availability is resumable and the total weighted completion time as objective.

As we can see in Table 3, the problem $1||\epsilon(\sum w_j^A C_j^A/\sum w_j^B C_j^B)$ is NP-hard. This complexity also holds for more than two sets of jobs (Lee et al., 2009). These authors provide an FPTAS to solve the goal programming version of this problem for the single machine and the unrelated parallel machines. In the same way, Agnetis et al. (2004) find $1||\epsilon(\sum U_j^1/\sum U_j^2, \ldots, \sum U_j^K)$ to be polynomially solvable since $1||\epsilon(\sum U_j^A/\sum U_j^B)$ is polynomially solvable too. This result implies that the goal programming counterpart is polynomial too, although the weighted version $1||GP(\sum w_j^1 U_j^1, \ldots, \sum w_j^K U_j^K)$ is proved to be strongly NP-complete by Cheng et al. (2006). These authors present an FPTAS for the problem and prove that it is polynomial when K is arbitrary and the weights and epsilon values of all sets are positive integers. Related to

the latter problem, Agnetis et al. (2007a,b) obtain a $\mathcal{O}(\sum_{k=1}^{K'} n^k \prod_{k=1}^{K} n^k + n \log n)$ running time algorithm for the problem $1||GP(\sum w_j^1 U_j^1, \ldots, \sum w_j^{K'} U_j^{K'}, \max f_j^{K'+1}, \ldots, \max f_j^K)$.

Regarding problems considering non-regular performance measures, Mor and Mosheiov (2010) study the general case where $\max f_j^k = \max_{j \in \mathcal{J}^k}\{f_j^k(E_j^k)\}$, with $k = A, B$, providing an algorithm to solve the problem in polynomial time with $\mathcal{O}(n^B \log n^B + (n^A)^2)$.

With respect to other layouts different from the single machine, although $1||\epsilon(\max C_j^A / \max C_j^B)$ is trivial, the ϵ-constraint problem with the same objective is shown to be NP-hard for even the 2-machine open shop (Agnetis et al., 2004) and, consequently, for the permutation flowshop (Luo et al., 2011). For the last problem, Luo et al. (2011) provide an FPTAS. A consequence is that $PFm||\epsilon(\max C_j^A / \max T_j^B)$ is NP-hard, and the structure of the solutions is studied by Perez-Gonzalez et al. (2011) for common and different due dates. $PF2||\epsilon(\sum C_j^A / \sum U_j^B)$ and $PF2||\epsilon(\sum T_j^A / \sum U_j^B)$ are both NP-hard (Lee et al., 2011, 2010). The authors present B&B as well as SA algorithms for both problems.

Finally, there are some rescheduling problems where the objective is to minimize a function of one set of jobs subject to that one or more functions of the other set to be bounded. All of them have the form $1||\epsilon(f^A/f_1^B, f_2^B)$, and are presented in Table 4. Some properties about feasibility and conditions for optimal schedules are given by Mu and Hao (2010) for $1||\epsilon(f^A/f^B, \max D_j^B)$ and $1||\epsilon(f^A/f^B, \max \Delta_j^B)$ when $f^A, f^B \in \{\max C_j, \sum C_j, \max L_j\}$. Polynomially solvable algorithms for $1||\epsilon(\max C_j^A / \max C_j^B, \max D_j^B)$ and $1||\epsilon(\max C_j^A / \max C_j^B, \max \Delta_j^B)$ are given by Mu and Gu (2010). The rest of the problems can be solved by dynamic programming algorithms (Mu and Gu, 2010; Mu and Hao, 2010).

	f_2^B			
	$\max D_j$	$\sum D_j$	$\max \Delta_j$	$\sum \Delta_j$
$1\|\|\epsilon(\max C_j^A / \max C_j^B, f_2^B)$	$\mathcal{O}(n^A \log n^A)$	$\mathcal{O}((n^A)^2(n^B)^2\epsilon_1)$	$\mathcal{O}(n^A \log n^A)$	$\mathcal{O}((n^B)^2(n^A)^2 p^A \epsilon_1)$
$1\|\|\epsilon(\sum C_j^A / \sum C_j^B, f_2^B)$	$\mathcal{O}(n^B(n^A)^2\epsilon_1)$	$\mathcal{O}((n^B)^2(n^A)^2\epsilon_1)$	$\mathcal{O}(n^B n^A p^A \epsilon_1 + n^A \log n^A)$	$\mathcal{O}(n^B(n^A)^2 p^A \epsilon_1)$

Table 4: Complexity of the algorithms to solve basic multiagent rescheduling problems on a single machine for LCC approach with the form $1||\epsilon(f^A/f_1^B, f_2^B)$

5.1.3 Pareto approach

In this section, we review the problem of finding the set of Pareto optima when there are disjoint sets of jobs. As discussed in Section 4, some authors only provide an algorithm to determine the size of the non-dominated sets. For $1||\#(\max F_j^A, \sum C_j^B)$, the size of the non-dominated set is at most $n^A n^B$ (Agnetis et al., 2004, 2007a). Agnetis et al. (2004, 2007b) provide procedures to determine the size of the non-dominated set with complexity $\mathcal{O}(n^A \log n^A + n^B \log n^B)$. For the weighted case, $1||\#(\max F_j^A, \sum w_j^B C_j^B)$, Agnetis et al. (2007b) show that it may have an exponential number of non-dominated solutions with respect to the instance size. The case with parallel machines $Pm||\#(\max C_j^A, \sum C_j^B)$, is shown to be NP-hard by Balasubramanian et al. (2009), developing several constructive heuristics and a GA. Saule and Trystram (2009) show that $Pm||\#(\max C_j^1, \ldots, \max C_j^K)$ is strongly NP-hard, and provide an approximation algorithm for the problem for an arbitrary number of machines.

Agnetis et al. (2004) prove that finding one non-dominated solution for $1||\#(\sum C_j^A, \sum C_j^B)$ is NP-hard, and the number of non-dominated solutions may be exponential with respect to the instance size. Saule and Trystram (2009) study the case with more than two sets, $1||\#(\sum C_j^1, \ldots, \sum C_j^K)$, proposing an approximation algorithm and extending the approach to $Pm||\#(\sum C_j^1, \ldots, \sum C_j^K)$. Lee et al. (2009) develop the same approximation algorithm for the goal programming counterparts $1||GP(\sum C_j^1, \ldots, \sum C_j^K)$ and $Rm||GP(\sum C_j^1, \ldots, \sum C_j^K)$. For the weighted case $1||\#(\sum w_j^A C_j^A, \sum w_j^B C_j^B)$, an algorithm based on the Nash Bargaining Solution is proposed by Agnetis et al. (2009b). For $1||\#(\sum w_j^1 C_j^1, \ldots, \sum w_j^K C_j^K)$ with K fixed, Lee et al. (2009) develop an FPTAS for computing a δ-efficient set (according to the definition given by Warburton, 1987), with δ the error bound. Additionally, Saule and Trystram (2009) consider the case for parallel machines, $Pm||\#(\sum w_j^1 C_j^1, \ldots, \sum w_j^K C_j^K)$, developing approximation algorithms for this problem and $Pm||\#(\sum w_j^1 C_j^1, \ldots, \sum w_j^{K'} C_j^{K'}, \max C_j^{K'+1}, \cdots, \max C_j^{K''})$ with $K = K' + K''$.

Finally, Agnetis et al. (2000) consider a job shop problem, $J2||\#(f^A, f^B)$, assuming that both $\mathcal{J}^A$ and $\mathcal{J}^B$ contain only one job with n^A and n^B operations, and f^A and f^B non-regular cost functions $f^i(C_i)$. For this problem, the set of nondominated schedules is identified, and an algorithm to generate them in polynomial time is provided. The running time depends on the number of incompatible pairs, defined as a pair of operations of $\mathcal{J}^A$ and $\mathcal{J}^B$ which require the same machine and cannot be done in parallel.

5.2 Non-disjoint (Interfering) sets of jobs

5.2.1 Linear Convex Combination approach

For this approach, only one machine problems have been studied. Table 5 shows the complexity of these problems with global objective for some regular measures analysed by Huynh-Tuong et al. (2011) and Huynh-Tuong and Soukhal (2010).

$f \backslash f^B$	$\max C_j$	$\max L_j$	$\sum C_j$	$\sum w_j C_j$	$\sum T_j$	$\sum w_j T_j$	$\sum U_j$	$\sum w_j U_j$
$\max C_j$	trivial	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	NP	NP	NP	NP
$\max L_j$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n^2)$	NP	NP	NP	NP	NP
$\sum C_j$	$\mathcal{O}(n \log n)$	Open	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	NP	NP	NP	NP
$\sum w_j C_j$	$\mathcal{O}(n \log n)$	NP	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	NP	NP	NP	NP
$\sum T_j$	NP	NP	NP	NP	NP	NP	NP	NP
$\sum w_j T_j$	NP	NP	NP	NP	NP	NP	NP	NP
$\sum U_j$	$\mathcal{O}(n \log n)$	Open	NP	NP	NP	NP	NP	NP
$\sum w_j U_j$	NP	NP	NP	NP	NP	NP	NP	NP

Table 5: Complexity for basic MASPs with global objective function on a single machine for LCC approach (Huynh-Tuong et al., 2011; Huynh-Tuong and Soukhal, 2010)

Hall and Potts (2004) study several rescheduling problems, shown in Table 6. They consider $\mathcal{J}^A$ as the set of old jobs, and $\mathcal{J}$ is the set of old and new jobs. Objective functions are $f \in \{\sum C_j, \max L_j\}$ and $f^A \in \{\max D_j^A, \sum D_j^A, \max \Delta_j^A, \sum \Delta_j^A\}$, considering all of them with a single disruption. Additionally,

the study includes the case with multiple disruption (md) for the problem $1||F_l(\sum C_j, \max D_j^A)$. Finally, $1||F_l(\max w_j T_j, \sum F_j^A)$ is addressed by Li and Cheng (1999), with $\mathcal{J}^A$ the set of the new jobs. The objective is to determine the due dates of the new jobs which are calculated depending on their completion times by considering $\sum F_j^A = \sum_{j \in \mathcal{J}^A} d_j(C_j)$ as a function cost. The problem is shown to be NP-hard, although the unweighted version is polynomially solvable in $\mathcal{O}(n^4)$.

$f \backslash f^A$	$\max D_j$	$\sum D_j$	$\max \Delta_j$	$\sum \Delta_j$
$\sum C_j$	$\mathcal{O}(n + n^A \log n^A)$; (md) $\mathcal{O}(n^A n^3)$	$\mathcal{O}(n^B \log n^B)$	$\mathcal{O}(n + n^B \log n^B)$	$\mathcal{O}(n^B \log n)$
$\max L_j$	$\mathcal{O}(n + n^B \log n^B)$	NP	$\mathcal{O}(n + n^B \log n^B)$	NP

Table 6: Complexity for basic multiagent rescheduling problems with global objective function on a single machine for LCC approach (Hall and Potts, 2004)

5.2.2 ϵ-constraint approach

Similar to the LCC approach, the single machine layout is the most extended case, all of them with global objective function. Taking into account that the problem is not symmetric, most of them have the form $1||\epsilon(f/f^A)$, whereas $1||\epsilon(f^A/f)$ has not been tackled in the literature. Table 7 shows the complexity of these problems for some regular measures given by Huynh-Tuong et al. (2011) and Huynh-Tuong and Soukhal (2010). Additionally, the problem with more than two sets of jobs $1||\epsilon(\max F_j / \max F_j^1, \ldots, \max F_j^K)$ is polynomially solvable in $\mathcal{O}(n^2)$ (Sadi et al., 2012). For this approach, Hall and Potts (2004) study some rescheduling problems whose complexities are shown in Table 8. All problems are studied with a single disruption, and for several cases where there may be several job arrivals, multiple disruptions (md) are included. Finally, Sadi et al. (2012) show that $Pm||\epsilon(f/f^A)$ is NP-hard if $f^A, f \in \{\max L_j, \sum C_j, \sum w_j C_j, \sum U_j, \sum w_j U_j, \sum T_j, \sum w_j T_j\}$. They also give a dynamic programming algorithm to solve the problem for $m = 2$ and $f^A, f \in \{\max C_j, \sum C_j\}$ in running time $\mathcal{O}(n^2 \epsilon)$.

$f \backslash f^B$	$\max C_j$	$\max L_j$	$\sum C_j$	$\sum w_j C_j$	$\sum T_j$	$\sum w_j T_j$	$\sum U_j$	$\sum w_j U_j$
$\max C_j$	trivial	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	NP	NP	NP	Open	NP
$\max L_j$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n)$	NP	NP	NP	Open	NP
$\sum C_j$	$\mathcal{O}(n \log n)$	Open	NP	NP	NP	NP	NP	NP
$\sum w_j C_j$	$\mathcal{O}(n \log n)$	NP	NP	NP	NP	NP	NP	NP
$\sum T_j$	NP	NP	NP	NP	NP	NP	NP	NP
$\sum w_j T_j$	NP	NP	NP	NP	NP	NP	NP	NP
$\sum U_j$	$\mathcal{O}(n^2)$	NP	NP	NP	NP	NP	Open	NP
$\sum w_j U_j$	NP	NP	NP	NP	NP	NP	NP	NP

Table 7: Complexity for basic MASPs with global objective function on a single machine for ϵ-constraint approach (Huynh-Tuong et al., 2011; Huynh-Tuong and Soukhal, 2010)

$f\backslash f^A$	$\max D_j$	$\sum D_j$	$\max \Delta_j$	$\sum \Delta_j$
$\sum C_j$	$\mathcal{O}(n + n^A \log n^A)$; (md) $\mathcal{O}(n^3)$	$\mathcal{O}((n^A n^B)^2)$	$\mathcal{O}(n + n^A \log n^A)$	s-NP
$\max L_j$	$\mathcal{O}(n + n^A \log n^A)$	Open	$\mathcal{O}(n + n^A \log n^A)$; (md) NP	NP; (md) NP

Table 8: Complexity for basic multiagent rescheduling problems with global objective function on a single machine for ϵ-constraint approach (Hall and Potts, 2004)

5.2.3 Pareto approach

In this case, we are only aware of the work by Ben Ltayef et al. (2009), who address the $PF2||\#(\max C_j, f^A)$ and $PF2||\#(\max T_j, f^A)$ problems, where $f^A \in \{\sum D_j^A, \max D_j^A, \sum \Delta_j^A, \max \Delta_j^A\}$. They give the mathematical formulation for these problems, and present solution procedures based on SA and TS.

6 Extended problems

Extended problems have some condition/s for jobs and/or machines, indicated in the field β. In the next subsections, we review the main contributions for each approach. Note that, given the heterogeneity of the problems addressed, it is not possible to clearly classify the contributions as in the case of basic problems. Nevertheless, we have identified the following categories: Release times and/or preemption; Variable processing times (processing times are not fixed values); Batch and/or set-up; and Others. These categories should be interpreted in a loose manner and only as a way to classify the disparity of contributions.

6.1 Disjoint (Competing) sets of jobs

6.1.1 Linear Convex Combination approach

Release times and/or preemption

Regarding problems in which release times are allowed, $1|r_j|F_l(\max C_j^A, \max C_j^B)$ is shown to be NP-hard by Ding and Sun (2010), who also give an optimal algorithm with $\mathcal{O}(n^A + n^B)$ running time for its preemptive version. For the on-line case, i.e. $1|\textit{on-line}, r_j, \textit{prmp}|F_l(\max C_j^A, \max C_j^B)$ they develop an approximation algorithm. The equivalent problem without preemption, $1|\textit{on-line}, r_j|F_l(\max C_j^A, \max C_j^B)$, is also addressed giving an on-line algorithm with a lower bound of the competitive ratio of the problem.

Different contributions are due to Cheng et al. (2008) for some specific cases with more than two sets and regular performance measures. They prove that $1|\textit{prec}, I, \textit{prmp}|F_l(F^1, \ldots, F^K)$ and $1|\textit{prec}|F_l(F^1, \ldots, F^K)$ can be solved in pseudo-polynomial time when K is fixed. Finally, the case without precedence $1|I, \textit{prmp}|F_l(F^1, \ldots, F^K)$ is polynomial with $\mathcal{O}((n^* \log n^* + n \log n)(\prod_{k=1}^K n_k)^K)$, where n^* is the number of forbidden intervals, when $F^k \in \{\max w_j^A C_j^k, \max L_j^k\}$.

6.1.2 ϵ-constraint approach

Release times and/or preemption

The assumption of preemption for all jobs in both sets has only been studied for the problem $1|prmp|\epsilon(\sum U_j^A/\max F_j^B)$ by Agnetis et al. (2004), who show that it is solvable in $\mathcal{O}(n^A \log n^A + n^B \log n^B)$. The case $1|prmp, r_j|\epsilon(\max F_j^A/\max F_j^B)$ where release times for all sets of jobs are added to the preemption hypothesis is addressed by Leung et al. (2010),who show that is polynomially solvable in $\mathcal{O}(n^2)$. This complexity can be further reduced to $\mathcal{O}(n^A \log n^A + n^B \log n^B)$ when the criterion for $\mathcal{J}^A$ is $\max L_j^A$ or $\sum C_j^A$, and to $\mathcal{O}(n^3(n^A n^B)^2)$ when it is $\sum U_j^A$. Moreover, they show that $1|prmp^A, r_j|\epsilon(\sum C_j^A/\max T_j^B)$ and $1|prmp^A, r_j|\epsilon(\max L_j^A/\max T_j^B)$ are solvable in $\mathcal{O}((n^A)^{n^B+1}\log n^A)$ when $\epsilon = 0$. Finally, they study some problems on parallel machines: $P2|prmp|\epsilon(\sum C_j^A/maxF_j^B)$ is solvable in $\mathcal{O}(n^A \log n^A + n^B \log n^B)$ (this result is shown by Wan et al. (2010) too). However, the problem with release dates for all jobs is NP-hard when $m \geq 2$ and fixed. Additionally, $Pm|prmp|\epsilon(\sum U_j^A/\max F_j^B)$ is solvable in $\mathcal{O}(n^2 n^A n^B)$ for $m = 2$, in $\mathcal{O}(n^{3m-5}(n^A)^2(n^B)^2)$ time for $m > 2$ and fixed, and NP-hard when m is arbitrary. Furthermore, they show that the assumption of release dates transform the problem into NP-hard even for $m \geq 2$ and fixed. Finally, the problem $Pm|prmp|\epsilon(\max F_j^A/\max F_j^B)$ is solvable in $\mathcal{O}(\log(\alpha_U - \alpha_L)n\log(nm))$, where $\alpha_U = \max_{j\in\mathcal{J}^A}\{F_j^A(P)\}$ and $\alpha_L = \min_{j\in\mathcal{J}^A}\{F_j^A(0)\}$. When release times are considered, the complexity is $\mathcal{O}(\log(\alpha_U - \alpha_L)n^3)$.

Leung et al. (2010) show that $1|r_j^B|\epsilon(\max F_j^A/\max T_j^B)$ is NP-hard. Note that this result, although developed independently from Ding and Sun (2010), is a consequence of the NP-hardness of the $1|r_j|F_l(\max C_j^A/\max C_j^B)$ shown by the latter authors. The problem is NP-hard too for the criteria $\sum C_j^A$ and $\sum U_j^A$. However, if preemption is allowed, all corresponding problems are polynomially solvable.

Yin et al. (2012b) consider two problems with release dates for all jobs. First, they prove that $1|r_j|\epsilon(\sum C_j^A/\max L_j^B)$ is strongly NP-hard. Therefore, $1|r_j|\epsilon(\sum T_j^A/\max L_j^B)$ is strongly NP-hard too, and they present a mixed integer programming model, a B&B with some dominance rules and a marriage in honey-bees optimization algorithm.

Cheng et al. (2007) study different problems with preemption, precedence constraints and forbidden intervals. They show that the problems $1|prmp, I, prec^B|\epsilon(\sum F_j^A/\max F_j^B)$ can be polynomially reduced to $1|prmp, I|\sum F_j^A$ in $\mathcal{O}((n^B)^2 + n^*\log n^*)$, solving it in $(\mathcal{O}((n^B)^2 + n^A\log n^A + n^*\log n^*))$ when $\sum F_j^A \in \{\sum C_j^A, \sum U_j^A\}$.

Finally, for more than two sets of jobs, Agnetis et al. (2007b) address $1|prmp|\epsilon(F^1/\max F_j^2, \ldots, \max F_j^K)$ and $1|prmp, I|\epsilon(F^1/\max F_j^2, \ldots, \max F_j^K)$ in order to discover some properties that were later applied to their basic counterpart. $Pm|prmp|\epsilon(\sum F_j/\sum F_j^1, \ldots, \sum F_j^K)$ is considered by Sadi et al. (2012), showing that it is NP-hard. They present a linear programming model for the general case with un-related machines in parallel.

Variable processing times

The references reviewed in this section assume that the processing times are not fixed in advance. They include deteriorating jobs, learning effect and controllable processing times. For the $1|deteriorating|\epsilon(\max F_j^A/\max F_j^B)$ problem, Yin et al. (2012a) give a polynomial time algorithm with running time $\mathcal{O}((n^A)^2 + n^B\log n^B)$. $1|deteriorating|\epsilon(\max L_j^A/\max C_j^B)$ and linear deterioration is shown to be $\mathcal{O}(n^A\log n^A)$ by Liu and Tang (2008); Liu et al. (2010a, 2011b).

There are no such general results when both functions are not under the max form. However,

there are algorithms to optimally solve a number of problems in $\mathcal{O}(n^A \log n^A + n^B \log n^B)$, such as $1|deteriorating|\epsilon(\sum C_j^A / \max F_j^B)$ with linear deterioration depending on the starting time (Liu and Tang, 2008; Liu et al., 2011a,b) and with linear deterioration depending on the position of the jobs (Liu et al., 2010c); as well as for $1|deteriorating|\epsilon(\sum E_j^A / \max F_j^B)$ and $1|deteriorating, d_j^A = d|\epsilon(\sum w_j^A E_j^A / \max F_j^B)$ (Yin et al., 2012a).

The problem $1|deteriorating|\epsilon(\sum T_j^A / \sum U_j^B)$ with $\epsilon = 0$ is tackled by Gawiejnowicz et al. (2010) and Gawiejnowicz et al. (2011). In the first reference, the authors develop a B&B combined with simple heuristics and local search, while in the second present an evolutionary algorithm.

Liu et al. (2010b) study the problem $1|deteriorating, s\text{-}batch|\epsilon(\sum C_j^A / \max F_j^B)$, where the processing times as well as the setup times of each job are linear functions of its starting time. They tackle two problems depending on the type of the deteriorating function, proposing two algorithms to solve both problems in $\mathcal{O}(n^A \log n^A)$ time respectively.

Regarding the learning effect, $1|learning|\epsilon(\sum C_j^A / \max F_j^B)$ is shown to be solvable in $\mathcal{O}(n^A \log n^A + n^B \log n^B)$ if there is a linear deterioration depending on the position of the jobs (Liu et al., 2010c). For the $1|learning|\epsilon(\sum T_j / \sum U_j)$ problem with $\epsilon = 0$, Wu et al. (2011a) present a B&B and develop heuristic algorithms for larger instance sizes. The problem $1|learning|\epsilon(\sum w_j^A C_j^A / \sum U_j^B)$ with $\epsilon = 0$ is addressed by Cheng et al. (2011a) with a position based function. They show that the problem is NP-hard since the corresponding basic case is already NP-hard (Ng et al., 2006), and propose a B&B and three SA-based algorithms.

The combination of position-based- learning and deteriorating effects is addressed by Cheng et al. (2011b), as they show that the $1|learning^A, deteriorating^B|\epsilon(\sum C_j^A / \max L_j^B)$ problem is NP-hard. A B&B is proposed, as well as several SA-based algorithms. Wu et al. (2011b) address the same problem with a different position-based function. Again, a B&B and metaheuristics based on ACO are provided and compared.

Regarding controllable processing times, Wan et al. (2010) address several $1|ctrl|\epsilon(f^A / f^B)$ problems:

- For the objective $\epsilon(Ctrl_{sum}^A / \max L_j^A, \max F_j^B)$ with $\epsilon = (0, \epsilon^B)$, the problems where β is $ctrl^A, r_j^A, prmp$; $ctrl^A, r_j, prmp$; and $ctrl^A, prmp^A$ are shown to be polynomial with running time $\mathcal{O}(n(\log n + k + 1))$, with k the number of different costs of $\mathcal{J}^A$. However, the case where β is $ctrl^A, r_j^A, prmp^A$ is strongly NP-hard.

- When β is $ctrl^A$ and $ctrl^A, prmp^A$, the problem with objective $\epsilon(Ctrl_{sum}^A + \sum C_j^A / \max F_j^B)$ is polynomial with running time $\mathcal{O}((n^A)^2 n^B n \log n)$, and $\mathcal{O}((n^A)^2 n^B n)$ for $\epsilon(Ctrl_{sum}^A + \max T_j^A / \max F_j^B)$ and $\epsilon(Ctrl_{sum}^A + \max L_j^A / \max F_j^B)$. However, when β is $ctrl^A, r_j^A, prmp^A$ all of them are strongly NP-hard.

Finally, for the parallel machines case, the problem $P2|ctrl^A, prmp|\epsilon(\sum F_j^A + \sum C_j^A / \max F_j^B)$ is shown to be polynomially solvable in $\mathcal{O}(n^B \log n^B + (n^A)^3 + (n^A)^2 n^B)$ by Wan et al. (2010).

Batch and/or set-up

The *p-batch* case has been studied by Li and Yuan (2011) for different forms of f^A and f^B, including both incompatible and compatible sets of jobs. Table 9 shows the complexities and the running times in the pseudo-polynomial and polynomial cases on a single machine, where $\alpha_U = \max_{j \in \mathcal{J}^A}\{f_j^A(P)\}$, $P = \sum_{j \in \mathcal{J}} p_j$ and $\alpha_L = \min_{j \in \mathcal{J}^A}\{f_j^A(0)\}$.

	$\epsilon(\max F_j^A / \max F_j^B)$	$\epsilon(\sum F_j^A / \max F_j^B)$	$\epsilon(\sum F_j^A / \sum F_j^B)$	$\epsilon(\sum w_j^A C_j^A / \max C_j^B)$
CS	$\mathcal{O}(n \log n \log(\alpha_U - \alpha_L))$	$\mathcal{O}(n^2 P)$	$\mathcal{O}(n^2 \epsilon P)$	NP
IS	$\mathcal{O}(nn^A n^B \log(\alpha_U - \alpha_L))$	$\mathcal{O}(nn^A n^B P)$	$\mathcal{O}(nn^A n^B \epsilon P)$	NP
	$\epsilon(\sum w_j^A C_j^A / \sum C_j^B)$	$\epsilon(\max L_j^A / \max F_j^B)$	$\epsilon(\sum U_j^A / \sum U_j^B)$	$\epsilon(\sum U_j^A / \max F_j^B)$
CS	NP	$\mathcal{O}(n \log n \log P)$	$\mathcal{O}(n^2 n^A n^B)$	$\mathcal{O}(n^2 n^A)$
IS	NP	$\mathcal{O}(nn^A n^B P)$	$\mathcal{O}(n^2 (n^A)^2 (n^B)^2)$	$\mathcal{O}(n^2 (n^A)^2 n^B)$

Table 9: Complexity of ϵ-constraint approach for extended MASPs on a single machine with $p - batch$

The *s-batch* case is addressed by Mor and Mosheiov (2011) when set-up times depend on the set and the batches of jobs in $\mathcal{J}^B$ must be processed continuously, i.e. the batches of jobs in $\mathcal{J}^A$ are partitioned into two sequences, scheduled prior to and after the batches of jobs in $\mathcal{J}^B$, respectively. They introduce an efficient $\mathcal{O}(n^{\frac{3}{2}})$ solution algorithm for the resulting problem $1|s\text{-}batch, p_j = 1|\epsilon(\sum C_j^A / \sum C_j^B)$.

The problem $1|s\text{-}batch|\epsilon(\sum w_j^B C_j^B / \max T_j^A, setup, \sum D_j^A)$ is analysed by Unal et al. (1997) with $\epsilon = (0, 0, 0)$. The objective is to minimize the total weighted completion times of jobs in $\mathcal{J}^B$, integrating them into the schedule of jobs in $\mathcal{J}^A$ making any job in $\mathcal{J}^A$ tardy and without incurring any setup times or changing the relative sequence of the jobs in $\mathcal{J}^A$. They show that the problem is NP-hard, and that the specific case where $p_j = 1 \ \forall j \in \mathcal{J}$ is solvable in polynomial time. This case turns to be strongly NP-hard if there are chain-like precedence constraints on the jobs in $\mathcal{J}^A$.

Others

Fan (2010) study the $1|V(1, \infty), direct|\epsilon(\sum C_j^A / \sum C_j^B)$ problem for a particular case where each set of jobs belongs to a customer situated at different locations. Jobs are processed and delivered to each customer by a uncapacitated vehicle. In this case, C_j is the completion time plus transportation time. $V(1, \infty), direct$ specifies that a single delivery vehicle can deliver any number of orders, and that only jobs going to the same customer can be delivered together in the same shipment. The authors prove that the problem is NP-hard, and develop a dynamic programming algorithm with complexity $\mathcal{O}(n^8)$.

In the case of unrelated machines, the only contributions are Elvikis et al. (2009) and Elvikis et al. (2010a) for the $Qm|p_j = p|\epsilon(F^A, \max C_j^B)$ problem, which is used to solve the Parero counterpart. The complexity of this problem depends on the form of F^A, and it is reduced from the complexity of the Pareto counterpart indicated in the following subsection.

Regarding flowshops, there are very few contributions. Huynh-Tuong and Soukhal (2009a) show that the $F3|\mathcal{J}^A(m_1 \mapsto m_2), \mathcal{J}^B(m_2 \mapsto m_3)|\epsilon(\max C_j^B / \max C_j^A)$ problem is NP-hard. Huynh-Tuong and Soukhal (2009b) provide a pseudo-polynomial algorithm with running time $\mathcal{O}(n^B \epsilon \beta)$, with $\beta = \sum_{j \in \mathcal{J}^B} (p_{2j} + p_{3j})$. Finally, Perez-Gonzalez et al. (2011) analyze the structure of solutions for the problem $PFm|d_j = d|\epsilon(\max C_j^A / \max T_j^B)$ when $\epsilon = 0$. As the problem is NP-hard by reduction from $PF2||\epsilon(\max C_j^A / \max C_j^B)$, Perez-Gonzalez and Framinan (2010b) develop some heuristics and a VNS method are proposed which are tested for this problem.

6.1.3 Pareto approach

Preemption and/or release dates

Cheng et al. (2007) present a polynomial algorithm to obtain the set of nondominated solutions for $1|prmp, I, prec^B|\#(f^A/\max f_j^B)$ with $f^A \in \{\sum C_j, \sum U_j\}$, giving a bound of the running time. The algorithm is valid for the cases without precedence constraints.

The problem $1|r_j, prmp|\#(\sum C_j^A, \sum w_j^B U_j^B)$ is shown to be NP-hard by Meiners and Torng (2007). However, they show that if $p_j = 1$ is polynomially solvable.

Regarding other settings, only Peha (1995) models the problem $Pm|r_j, p_j = 1|\#(\sum w_j^A C_j^A, \sum w_j^B U_j^B)$ from a practical application about real-time systems and integrated services networks. They develop an algorithm to solve this problem in $\mathcal{O}(n^2)$ running time.

Batch and/or set-up

Tan et al. (2011) consider the $1|p\text{-}batch|\#(\max C_j^A, \max C_j^B)$ problem where each job has a different size. This problem is shown to be NP-hard by reduction to the case with one set of jobs. They give the mathematical formulation of the problem, and provide an ACO to solve it. Moreover, an evaluation is carried out in order to compare ACO to a GA adapted from the one set problem.

Yazdani Sabouni and Jolai (2010) provide one heuristic for each of the following problems: Problem $1|p\text{-}batch, IS, b \leq n|\#(\max C_j^A, \max L_j^B)$ and equal size of the jobs, problem $1|p\text{-}batch, IS|\#(\max C_j^A, \max L_j^B)$, and problem $1|p\text{-}batch, CS, b \leq n, p_j^B = p|\#(\max C_j^A, \max L_j^B)$ and equal size of the jobs. They also show that the heuristic for the first problem turns to be optimal if $p_j^B = p$. Finally, Feng et al. (2011) study the problem $1|s\text{-}batch, IS|\#(\max C_j^A, \max L_j^B)$, giving a polynomial time algorithm with $\mathcal{O}(n^A + (n^B)^4)$.

Others

The case of unrelated machines has been only studied in the context where $p_j = p$ when $\mathcal{J}^A$ and $\mathcal{J}^B$ have only one job with n^A and n^B operations respectively, $Qm|p_j = p|\#(\max F_j^A, \max F_j^B)$, addressed by Elvikis and T'kindt (2012); Elvikis et al. (2010b). They develop an algorithm that enumerates all Pareto solutions of this problem in $\mathcal{O}((n^A)^2 n^B + n^A n^B \log n^B)$ running time. The algorithms are further refined by Elvikis et al. (2009, 2010a) for $Qm|p_j = p|\#(F^A, \max C_j^B)$, with F^A any regular function. For this problem, they develop an algorithm to enumerate the set of Pareto solutions by solving iteratively the ϵ-constraint version of this problem with different ϵ values. The algorithm has $\mathcal{O}(n \log m + n^A \psi)$ running time, with ψ depending on the time needed to assign jobs in $\mathcal{J}^A$. When F^A takes a regular sum form, the aforementioned generic algorithm reduces its running time to $\mathcal{O}(n^4)$. Moreover, for $\sum w_j^A C_j^A$, $\sum T_j^A$ and $\sum w_j^A T_j^A$ with the due dates and weights agreeable, i.e. $d_j \leq d_{j'} \Rightarrow w_j \geq w_{j'}$, the running time is $\mathcal{O}(n \log m + \psi')$, with ψ' the time to solve $1|p_j = 1|\sum F_j^A$. Additionally, for $\sum U_j^A$ and $\sum w_j^A U_j^A$, the running time is $\mathcal{O}(n \log m + n^A \log n^A)$. When F^A takes a regular max form, the nondominated solutions can be found in $\mathcal{O}(n^3)$. Furthermore, for $\max C_j^A$, there are only two strictly non-dominated solutions, and finally, for $\max L_j^A$ they develop an $\mathcal{O}(n \log m + n^A \log n^A)$ running time algorithm.

Finally, Peha (1995) consider the problem $Pm|p_j = 1|\#(\sum w_j^A C_j^A, \sum w_j^B U_j^B)$ where the complexity of the algorithm is $\mathcal{O}(n \log n)$.

6.1.4 Other problems

Some references found in the literature consider MASPs with different objective functions and do not correspond to the previous subsections. Some of them are applied problems.

Agnetis et al. (2007a) consider a constraint specific of MASPs, where a coordination protocol is defined by an arbitrator. Each set of jobs belongs to an agent, who submits the jobs one by one to the arbitrator on each step. Between those jobs provided for each agent, the arbitrator selects one according to a priority rule, and schedules it at the end of the current schedule (initially empty). In this problem, the objective is to determine the order to provide the jobs for each agent to the arbitrator, since the priority rule is public information, whereas jobs characteristics are private information of the respective agent. Some rules encountered in the literature include priority rules (such as the well-known SPT, $WSPT$, EDD); Round-Robin rule (denoted as RR), where jobs are alternated from each set; or k-$\mathcal{R}$, where at most k consecutive jobs of the same set are selected according to rule $\mathcal{R}$. Considering the single machine problem, Agnetis et al. (2007a) prove that $1|RR|F_l(\sum C_j^A, \sum C_j^B)$, $1|RR|\#(\sum C_j^A, \sum C_j^B)$, $1|k\text{-}\mathcal{R}|F_l(\sum C_j^A, \sum C_j^B)$ and $1|k\text{-}\mathcal{R}|\#(\sum C_j^A, \sum C_j^B)$ are solvable using the SPT rule. Although $1|WSPT|\#(\sum w_j^A C_j^A, \sum w_j^B C_j^B)$ is optimally solvable by WSPT rule, for the LCC case the weights must be the same. Finally, they also prove that, for the problem $1|EDD|F_l(\sum U_j^A, \sum U_j^B)$, the coordination rule (for example EDD) turns out to be ineffective.

In the previous sections we have mentioned some problems considering the goal programming approach used to solve problems of different approaches. However, there are some references explicitly addressing a goal programming objective. For example, $1|p-batch, IS|GP(\max L_j^A, \max F_j^B)$ is solved by Li and Yuan (2011) by an algorithm developed previously to solve $1|p-batch, IS|\epsilon(\max F_j^A / \max F_j^B)$, and which can be easily applied to this problem. The running time of this algorithm is $\mathcal{O}(nn^A n^B \log(\alpha_U - \alpha_L))$ where $\alpha_U = \max_{j\in\mathcal{J}^A}\{F_j^A(P)\}$ with $P = \sum_{i=1}^{m}\sum_{j\in\mathcal{J}} p_{ij}$, and $\alpha_L = \min_{j\in\mathcal{J}^A}\{F_j^A(0)\}$. Moreover, Cheng et al. (2008) shows that $1|prec|GP(\max F_j^1, \ldots, \max F_j^K)$ is solvable in $\mathcal{O}(n^2)$, and the running time is $\mathcal{O}(n^* \log n^* + n^2)$, depending on the number of forbidden intervals, n^*, if we consider the case with preemption. The same problem is polynomially solvable in $\mathcal{O}(n^* \log n^* + n \log n)$ without the precedence constraint. Finally, Cheng et al. (2006) shows that $1|prmp|GP(\sum U_j^1, \ldots, \sum U_j^K)$ is equivalent to the basic case, and this problem with forbidden intervals is reducible in linear time to the equivalent basic problem, with modified due dates.

The rest of the references are applied problems. First, Peha and Tobagi (1990) consider a MASP applied to traffic in telecommunications (file transfer or interprocess communication) on a single packet-switched network. They consider two sets of jobs formed by packets, $\mathcal{J}^A$ and $\mathcal{J}^B$, and a lexicographical approach where the objective functions are $\frac{\sum w_j^A U_j^A}{\sum_{j\in\mathcal{J}^A} w_j^A}$ and $\frac{\max w_j^B L_j^B}{\sum_{j\in\mathcal{J}^B} w_j^B}$. They present an algorithm with $\mathcal{O}(n^2)$ for the weighted and the unweighted cases.

Arbib et al. (2004) consider two problems with two disjoint sets of jobs $\mathcal{J}^A$ and $\mathcal{J}^B$: in the first problem the objective is to maximize the minimum between $\sum F_j^A$ and $\sum F_j^B$. The second one is an ϵ-constraint approach where the objective is to maximize $\sum F_j^A$ subject to $\sum F_j^B \geq \epsilon$. Problems are solved by pseudo-polynomially dynamic programming algorithm.

Finally, some authors consider the problem where jobs (in $\mathcal{J}^A$) and maintenance tasks (in $\mathcal{J}^B$), are

scheduled together. PM^B is the frequency of the maintenance tasks, indicating that there is a tolerance interval between the earliest and the latest time separating two consecutive occurrences of the maintenance task $j \in \mathcal{J}^B$ for each machine. Bouzid-Sitayeb et al. (2008) develop an ACO and it is compared to a GA for the problem $PFm|PM^B|Lex(\max C_j^A, \sum E_j^B + \sum L_j^B)$. Moreover, Khelifati and Bouzid-Sitayeb (2011b,a) consider the same problem for the LCC approach, $PFm|PM^B|F_l(\max C_j^A, \sum E_j^B + \sum L_j^B)$, suggesting a multi-agent system and developing a simulation experiment.

6.2 No-disjoint (Interfering) set of jobs

6.2.1 Linear Convex Combination approach

Some rescheduling problems are studied by Yang (2007). They prove that $1|ctrl^B|F_l(f, Ctrl_{sum}^B, \sum \Delta_j^A)$ is polynomially solvable in $\mathcal{O}((n^A)^{2.5}(n^B)^{2.5})$ time when f is $\sum C_j$. However, when f is $\sum T_j$ the problem is NP-hard, and Yang (2007) solve it by a very large scale neighborhood search method, which is compared to a B&B algorithm.

6.2.2 ϵ-constraint approach

A number of results have been produced for the $1|r_j|\epsilon(\max C_j/f^A)$ problem for different functions f^A:

- For $\max D_j^A$, the problem is polynomially solvable in $\mathcal{O}((n^B)^2 n)$ (Yuan and Mu, 2007). The on-line version is solvable by algorithms with competitive ratio $\frac{3}{2}$ (Mu and Guo, 2009a,b).
- For $\sum D_j^A$, the problem is solvable in $\mathcal{O}((n \log n + (n^B)^2) \log r_{max})$ with $r_{max} = \max_{j \in \mathcal{J}} r_j$ (Yuan et al., 2007). They show that the symmetric problem $1|r_j|\epsilon(\sum D_j^A / \max C_j)$ can be solved in $\mathcal{O}(n \log n + (n^B)^2)$.
- For $\max \Delta_j^A$, it is strongly NP-hard (Yuan and Mu, 2007). However, Mu and Guo (2009b) show that the on-line version is solvable by an algorithm with competitive ratio $\frac{3}{2}$.
- For $\sum \Delta_j^A$, it is strongly NP-hard (Yuan and Mu, 2007).

$1|deteriorating|\epsilon(\sum C_j/f^A)$ is analysed by Zhao and Tang (2010) for different functions f^A:

- For $\max L_j^A$ is solvable in $\mathcal{O}(n \log n)$ when $\epsilon = 0$.
- Two algorithms of identical complexity $\mathcal{O}(n + n^B \log n^B)$ are provided for $\max D_j^A$, one in the case of job-dependent processing rate, and other where there is a constant processing rate for all jobs.
- Two algorithms are provided for $\max \Delta_j^A$, one in the case of job-dependent processing rate, and other where there is a constant processing rate for all jobs. The complexity of the first one is $\mathcal{O}(n+n^B \log n^B)$; being $\mathcal{O}(n^5)$ that of the second one.

Regarding learning effect, $1|learning|\epsilon(\sum w_j C_j / \max C_j^A)$ is addressed by Li and Hsu (2011). Since the corresponding basic problem is already NP-hard, they present some dominance properties and a lower bound for the B&B algorithm. Then, three GA algorithms are proposed and compared.

Finally, regarding batches, the rescheduling problem $1|s\text{-}batch, FR^A|\epsilon(\max C_j/\max D_j^A)$ is studied by Mocquillon et al. (2008). FR^A means that the order of jobs in $\mathcal{J}^A$ is given and it has to be kept. They propose a polynomial time algorithm with running time $\mathcal{O}(n + n^B \log_2 n^B)$.

7 Conclusions

In this work, we have reviewed a high number of papers containing different contributions to the field of multicriteria scheduling problems considering two or more sets of jobs. Up to 75 papers have been identified as related to the problem in the sense presented in this work, including only those papers written in English. Doing an analysis of the literature we concluded that the most suitable name for this kind of problems is interfering jobs scheduling problem. However, the most used term is multiagent scheduling problem (MASP). Problems have been classified in basic or extended problems, indicating the approach considered (Linear Convex Combination approach; ϵ-constraint and Pareto approach). In the following, we will highlight the main conclusions and try to set some possible future research avenues.

The first comment refers to the fact that there are much more works solely devoted to extended problems (47) than to basic problems (20). Moreover, eight references consider both types of problems. This may be seen as an indicator that the research has been so far conducted by specific application areas rather than for a systematic approach, which is not surprising given the lack of an integrated framework that hopefully our paper may help to build. While there are clear advantages for the application-oriented approach (such as providing sound, real-life justification for research in the area) more research on the basic problems would set the foundations for gaining further advances on extended problems. Then, extensions of the solution procedures for the former can be implemented in the latter.

A second general comment is that most of the works refer to the one machine setting, which is clearly an indication of the fact that we are on the early stages of the research. Moreover, most of the single machine problems are NP-hard, so we can expect that a big number of problems are NP-hard. Therefore, in the next years, the number of contributions addressing other settings would increase. Interestingly, the proportion of works dealing with layouts different than the one-machine is higher for the extended problems than for the basic problems. Again, this point out the direction that most research has been application-driven.

Regarding basic problems, the case of disjoint sets of jobs has been the one receiving more attention. Some comments about the main results can be done:

- In the one-machine layout, the LCC approach has been found to be polynomially solvable for many objectives, and only some specific cases of weighted objectives are shown not to be polynomial. Outside this layout, only the permutation flowshop with two machines has been studied for a very simple case ($\max C_j^A$ and $\max C_j^B$), which is already NP-hard.

- The ϵ-constraint approach has been found not to be polynomially solvable except for a very simple case, even for one machine and two objectives. However, relatively few approximate algorithms have been proposed for these problems. The cases with three or more objectives have been addressed only for specific rescheduling-related objective functions.

- With respect to the Pareto approach, less work has been done, bounding the number of non-dominated solutions for most of the problems, mainly because the complexity of this approach.

Basic problem with non disjoint sets of jobs have been addressed only for the case where one of the sets contains all jobs (called interfering jobs problems with global objective function). An interesting future research line could be to analyse the complexity for the more general case of non-disjoint (interfering) sets of jobs (i.e. when one of sets does not contain all jobs). The contributions with global objective function consider the ϵ-constraint and LCC approaches, and they are due to two groups of authors, analysing complexities for a great number of problems. There are hardly any algorithm proposed for these problems, so it is a open issue for this approach. In contrast, such algorithms have been proposed for the Pareto approach.

Regarding the extended problems, the case considering only one constraint has been scarcely studied, and a research avenue could be initiated in this area. Most of the works have been done for the case of disjoint sets of jobs under the LCC approach. In this type of problems, some general results have been presented regarding the learning effect. In contrast, those obtained for controllable processing times are very specific, focusing on the determination of the polynomially solvable cases. Regarding the Pareto approach, it is worth to note that many problems have been found not to be polynomially solvable, and several approximate algorithms (mostly based on metaheuristics) have been presented. Extended problems with non-disjoint sets of jobs have been scarcely studied, all of them for one machine.

Regarding solution procedures, there is an extensive contribution on studying the complexity, and a considerable number of one machine problems providing exact procedures have been found. It would be interesting to extend these results to other, more complex, machine layouts, even if it is likely that the complexity would increase. In contrast, there are relatively few approximate procedures available for the rest of the problems addressed with higher complexity (although it can be observed a concentration of similar methods for similar problems, for example some papers considering deteriorating and learning constraints). The relative scarcity of solution procedures is again another sign of a research field in its earliest stages, and it makes difficult to apply the research results. Another consequence of this fact is that there are only a handful of test beds to generate problem instances in which the solution procedures can be tested. Most of them consider the same size of the sets of jobs, avoiding extreme cases. Therefore, it may be interesting to generate a common and exhaustive test bed, considering different cases for the sizes of the jobs, in order to facilitate the comparison among solution procedures.

Finally, it is also worth mentioning that, in the case of more than two sets of jobs, most of the work is preliminary and for some specific cases, pointing at other future area of research.

Acknowledgements

The authors are sincerely grateful to the anonymous referees. This research has been funded by the Spanish Ministry of Science and Innovation, under the project "SCORE" with reference DPI2010-15573/DPI.

References

Agnetis, A., De Pascale, G., and Pacciarelli, D. (2009a). A lagrangian approach to single-machine scheduling problems with two competing agents. *Journal of Scheduling*, 12(4):401–415.

Agnetis, A., De Pascale, G., and Pranzo, M. (2009b). Computing the nash solution for scheduling bargaining problems. *International Journal of Operational Research*, 6(1):54–69.

Agnetis, A., Mirchandani, P. B., Pacciarelli, D., and Pacifici, A. (2000). Nondominated schedules for a job-shop with two competing users. *Computational and Mathematical Organization Theory*, 6(2):191–191.

Agnetis, A., Mirchandani, P. B., Pacciarelli, D., and Pacifici, A. (2004). Scheduling problems with two competing agents. *Operations Research*, 52(2):229–242.

Agnetis, A., Pacciarelli, D., and Pacifici, A. (2007a). Combinatorial models for multi-agent scheduling problems. In Levner, E., editor, *Multiprocessor Scheduling: Theory and Applications*, book chapter 2, pages 21–46. I-TECH Education and Publishing, Vienna (Austria).

Agnetis, A., Pacciarelli, D., and Pacifici, A. (2007b). Multi-agent single machine scheduling. *Annals of Operations Research*, 150(1):3–15.

Arbib, C., Smriglio, S., and Servilio, M. (2004). A competitive scheduling problem and its relevance to umts channel assignment. *Networks*, 44(2):132–141.

Baker, K. R. and Smith, J. C. (2003). A multiple-criterion model for machine scheduling. *Journal of Scheduling*, 6(1):7–16.

Balaji, P. G. and Srinivasan, D. (2010). An introduction to multi-agent systems. In *Innovations in Multi-Agent Systems and Applications*, book chapter 1, pages 1–27. Springer-Verlag Berlin Heidelberg 2010.

Balasubramanian, H., Fowler, J., Keha, A., and Pfund, M. (2009). Scheduling interfering job sets on parallel machines. *European Journal of Operational Research*, 199(1):55–67.

Ben Ltayef, A., Loukil, T., and Teghem, J. (2009). Rescheduling a permutation flowshop problems under the arrival a new set ofjobs. In *2009 International Conference on Computers and Industrial Engineering, CIE 2009*, pages 188–192.

Bouzid-Sitayeb, F., Ammi, I., Varnier, C., and Zerhouni, N. (2008). Applying ant colony optimization for the joint production and preventive maintenance scheduling problem in the flowshop sequencing problem. In *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, ICTTA*.

Cassady, C. R. and Kutanoglu, E. (2003). Minimizing job tardiness using integrated preventive maintenance planning and production scheduling. *IIE Transactions*, 35(6):503–513.

Cheng, T. C. E., Cheng, S. R., Wu, W. H., Hsu, P. H., and Wu, C. C. (2011a). A two-agent single-machine scheduling problem with truncated sum-of-processing-times-based learning considerations. *Computers and Industrial Engineering*, 60(4):534–541.

Cheng, T. C. E., Ng, C. T., and Yuan, J. J. (2006). Multi-agent scheduling on a single machine to minimize total weighted number of tardy jobs. *Theoretical Computer Science*, 362(1-3):273–281.

Cheng, T. C. E., Ng, C. T., and Yuan, J. J. (2007). Two-agent scheduling on a single machine with forbidden intervals and preemption. Technical report, Working paper, Department of Logistics, Hong Kong Polytechnic University.

Cheng, T. C. E., Ng, C. T., and Yuan, J. J. (2008). Multi-agent scheduling on a single machine with max-form criteria. *European Journal of Operational Research*, 188(2):603–609.

Cheng, T. C. E., Wu, W. H., Cheng, S. R., and Wu, C. C. (2011b). Two-agent scheduling with position-based deteriorating jobs and learning effects. *Applied Mathematics and Computation*, 217(21):8804–8824.

Coudert, T., Grabot, B., and Archimede, B. (2002). Production/maintenance cooperative scheduling using multi-agents and fuzzy logic. *International Journal of Production Research*, 40(18):4611–4632.

Ding, G. and Sun, S. (2010). Single-machine scheduling problems with two agents competing for makespan. *Lecture Notes in Computer Science*, 6328(PART 1):244–255.

Ding, G. and Sun, S. (2011). Single machine family scheduling with two competing agents to minimize makespan. *Asia-Pacific Journal of Operational Research*, 28(6):773–785.

Ehrgott, M. (2005). *Multicriteria Optimization*. Springer, 2^{a} edition.

Elvikis, D., Hamacher, H. W., and T'kindt, V. (2009). Scheduling two interfering job sets on uniform parallel machines with makespan and cost functions. Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA). Dublin, Ireland, pages 645–654.

Elvikis, D., Hamacher, H. W., and T'kindt, V. (2010a). Scheduling two agents on uniform parallel machines with-makespan and cost functions. *Journal of Scheduling*, 14(5):471–481.

Elvikis, D., Hamacher, H. W., and T'kindt, V. (2010b). Scheduling two interfering job sets on uniform parallel machines with maximum criteria functions. International Conference on Project Management and Scheduling, PMS. Tours, France, pages 179–182.

Elvikis, D. and T'kindt, V. (2012). Two-agent scheduling on uniform parallel machines with min-max criteria. *Annals of Operations Research*, In press.

Fan, J. (2010). Integrated production and delivery scheduling problem with two competing agents. volume 2 of *ICAMS 2010 - Proceedings of 2010 IEEE International Conference on Advanced Management Science*, pages 157–160.

Feng, D. and Yuan, J. J. (2007). Np-hardness of a multicriteria scheduling on two families of jobs. *OR Transactions*, 114:121–126.

Feng, Q., Yu, Z., and Shang, W. (2011). Pareto optimization of serial-batching scheduling problems on two agents. International Conference on Advanced Mechatronic Systems, ICAMechS 2011, pages 165–168.

Garey, M. R. and Johnson, S. M. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York.

Gawiejnowicz, S. (2008). *Time-Dependent Scheduling*. Monographs in Theoretical Computer Science. Springer, 1ª edition.

Gawiejnowicz, S., Lee, W. C., Lin, C. L., and Wu, C. C. (2010). A branch-and-bound algorithm for two-agent single-machine scheduling of deteriorating jobs. International Conference on Project Management and Scheduling, PMS. Tours, France, pages 207–211.

Gawiejnowicz, S., Lee, W. C., Lin, C. L., and Wu, C. C. (2011). Single-machine scheduling of proportionally deteriorating jobs by two agents. *Journal of the Operational Research Society*, 62(11):1983–1991.

Graham, R., Lawler, E. L., Lenstra, J., and Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326.

Hall, N. G. and Potts, C. N. (2004). Rescheduling for new orders. *Operations Research*, 52(3):440–453.

Herrmann, J. W. (2006). Rescheduling strategies, policies, and methods. In *Handbook of Production Scheduling*, volume 89 of *International Series in Operations Research & Management Science*, book chapter 6, pages 135–148. Springer.

Hoogeveen, H. (2005). Multicriteria scheduling. *European Journal of Operational Research*, 167(3):592–623.

Huynh-Tuong, N. and Soukhal, A. (2009a). Interfering job set scheduling on three-stage flowshop with a common stage machine. Technical report, Computer Science Laboratory of Tours University, France.

Huynh-Tuong, N. and Soukhal, A. (2009b). Interfering job set scheduling on two-operation three-machine flowshop. 2009 IEEE-RIVF International Conference on Computing and Communication Technologies: Research, Innovation and Vision for the Future, RIVF 2009.

Huynh-Tuong, N. and Soukhal, A. (2010). Single machine scheduling with interfering jobs. International Conference on Project Management and Scheduling, PMS. Tours, France, pages 243–247.

Huynh-Tuong, N., Soukhal, A., and Billaut, J. C. (2010). New scheduling problems with interfering and independent jobs. Technical report, Computer Science Laboratory of Tours University, France.

Huynh-Tuong, N., Soukhal, A., and Billaut, J. C. (2011). Single-machine multi-agent scheduling problems with a global objective function. *Journal of Scheduling*, 15(3):311–321.

Kellerer, H. and Strusevich, V. A. (2010). Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications. *Algoritmica*, 57(4):769–795.

Khelifati, S. L. and Bouzid-Sitayeb, F. (2011a). A multi-agent scheduling approach for the joint scheduling of jobs and maintenance operations in the flow shop sequencing problem. *Lecture Notes in Computer Science*, 6923 LNAI(PART 2):60–69.

Khelifati, S. L. and Bouzid-Sitayeb, F. (2011b). A sequantial distributed approach for the joint scheduling of jobs and maintenance operations in the flowshop sequencing problem. In *International Conference on Intelligent Systems Design and Applications, ISDA*, pages 420–425.

Lawler, E. L., Lenstra, J., Rinnooy Kan, A. H. G., and Shyu, S. J. (1993). Sequencing and scheduling: algorithms and complexitiy. In *Handbooks in Operations Research and Management Science, Volume 4: Logistics of Production and Inventory*, book chapter 9. Elsevier Science Publishers.

Lee, K., Choi, B. C., Leung, J. Y. T., and Pinedo, M. L. (2009). Approximation algorithms for multi-agent scheduling to minimize total weighted completion time. *Information Processing Letters*, 109(16):913–917.

Lee, W. C., Chen, S. K., Chen, C. W., and Wu, C. C. (2011). A two-machine flowshop problem with two agents. *Computers and Operations Research*, 38(1):98–104.

Lee, W. C., Chen, S. K., and Wu, C. C. (2010). Branch-and-bound and simulated annealing algorithms for a two-agent scheduling problem. *Expert Systems with Applications*, 37(9):6594–6601.

Leung, J. Y. T., Pinedo, M. L., and Wan, G. (2010). Competitive two-agent scheduling and its applications. *Operations Research*, 58(2):458–469.

Li, C. L. and Cheng, T. C. E. (1999). Due-date determination with resequencing. *IIE Transactions*, 31(2):183–188.

Li, D. C. and Hsu, P. H. (2011). Solving a two-agent single-machine scheduling problem considering learning effect. *Computers and Operations Research*, 39(7):1644–1651.

Li, S. and Yuan, J. (2011). Unbounded parallel-batching scheduling with two competitive agents. *Journal of Scheduling*, In press.

Liu, P., Feng, D., Zhou, X., and Tang, Q. (2010a). A note on two-agent single-machine scheduling problem with deteriorating jobs. Chinese Control and Decision Conference, CCDC 2010, pages 3832–3835.

Liu, P. and Tang, L. (2008). Two-agent scheduling with linear deteriorating jobs on a single machine. *Lecture Notes in Computer Science*, 5092:642–650.

Liu, P., Tang, L., and Zhou, X. (2010b). Two-agent group scheduling with deteriorating jobs on a single machine. *International Journal of Advanced Manufacturing Technology*, 47(5-8):657–664.

Liu, P., Yi, N., and Zhou, X. (2011a). A single-machine scheduling problem with two agents and decreasing linear deteriorating jobs. Proceedings of the 2011 Chinese Control and Decision Conference, CCDC 2011, pages 279–282.

Liu, P., Yi, N., and Zhou, X. (2011b). Two-agent single-machine scheduling problems under increasing linear deterioration. *Applied Mathematical Modelling*, 35(5):2290–2296.

Liu, P., Zhou, X., and Tang, L. (2010c). Two-agent single-machine scheduling with position-dependent processing times. *International Journal of Advanced Manufacturing Technology*, 48(1-4):325–331.

Luo, W., Chen, L., and Zhang, G. (2011). Approximation schemes for two-machine flow shop scheduling with two agents. *Journal of Combinatorial Optimization*, In press.

Meiners, C. R. and Torng, E. (2007). Mixed criteria packet scheduling. *Lecture Notes in Computer Science*, 4508:120–133.

Minella, G., Ruiz, R., and Ciavotta, M. (2008). A review and evaluation of multiobjective algorithms for the flowshop scheduling problem. *INFORMS Journal on Computing*, 20(3):451–471.

Mocquillon, C., Lenté, C., and T'kindt, V. (2008). Rescheduling for new orders with setup times. Eleventh International Workshop on Project Management and Scheduling, PMS 2008. Istanbul (Turkey), pages 203–205.

Mor, B. and Mosheiov, G. (2010). Scheduling problems with two competing agents to minimize minmax and minsum earliness measures. *European Journal of Operational Research*, 206(3):540–546.

Mor, B. and Mosheiov, G. (2011). Single machine batch scheduling with two competing agents to minimize total flowtime. *European Journal of Operational Research*, 215(3):524–531.

Mu, Y. and Gu, C. (2010). Rescheduling to minimize makespan under a limit on the makespan of the original jobs. volume 1 of *The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010*, pages 613–617.

Mu, Y. and Guo, X. (2009a). On-line rescheduling to minimize makespan nuder a limit on the maximum sequence disruption. In *Proceedings - 2009 IITA International Conference on Services Science, Management and Engineering, SSME 2009*, pages 479–482.

Mu, Y. and Guo, X. (2009b). On-line rescheduling to minimize makespan under a limit on the maximum disruptions. In *2009 International Conference on Management of e-Commerce and e-Government, ICMeCG 2009*, pages 141–144.

Mu, Y. and Hao, Y. (2010). Rescheduling to minimize the total completion time under a limit on the total completion time of the original jobs. In *Proceedings - 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2010*, volume 4, pages 307–311.

Ng, C. T., Cheng, T. C. E., and Yuan, J. J. (2006). A note on the complexity of the problem of two-agent scheduling on a single machine. *Journal of Combinatorial Optimization*, 12(4):386–393.

Nong, Q. Q., Cheng, T. C. E., and Ng, C. T. (2011). Two-agent scheduling to minimize the total cost. *European Journal of Operational Research*, 215(1):39–44.

Peha, J. M. (1995). Heterogeneous-criteria scheduling: Minimizing weighted number of tardy jobs and weighted completion time. *Computers and Operations Research*, 22(10):1089–1100.

Peha, J. M. and Tobagi, F. A. (1990). Evaluating scheduling algorithms for traffic with heterogeneous performance objectives. Global Telecommunications Conference, 1990, and Exhibition.'Communications: Connecting the Future', GLOBECOM '90., IEEE, pages 21–27.

Perez-Gonzalez, P. and Framinan, J. M. (2009). Scheduling permutation flowshops with initial availability constraint: Analysis of solutions and constructive heuristics. *Computers & Operations Research*, 36(10):2866–2876.

Perez-Gonzalez, P. and Framinan, J. M. (2010a). Finite-capacity due date setting in permutation flowshops: Analysis of solutions and simple heuristics. International Workshop on Project Management and Scheduling, PMS. Tours - Loire Valley, France, pages 319–322.

Perez-Gonzalez, P. and Framinan, J. M. (2010b). Setting a common due date in a constrained flowshop: A variable neighbourhood search approach. *Computers & Operations Research*, 37(10):1740–1748.

Perez-Gonzalez, P., Framinan, J. M., and Molina Pariente, J. M. (2011). Due date setting in permutation flowshop with arrival of new jobs. International Conference on Industrial Engineering and Systems Management, IESM. Metz, France.

Pinedo, M. (1995). *Scheduling: Theory, algorithms, and systems*. Prentice Hall International Series in Industrial and System Engineering. Prentice Hall, Englewood Cliffs, New Jersey (USA).

Ruiz, R., Carlos Garcia-Diaz, J., and Maroto, C. (2007). Considering scheduling and preventive maintenance in the flowshop sequencing problem. *Computers & Operations Research*, 34(11):3314–3330.

Sadi, F., Soukhal, A., and Billaut, J. C. (2012). Parallel machines multi-agent scheduling problems with a global objective function. International Conference on Project Management and Scheduling, PMS. Leuven, Belgium, pages 274–277.

Saule, E. and Trystram, D. (2009). Multi-users scheduling in parallel systems. IPDPS 2009 - Proceedings of the 2009 IEEE International Parallel and Distributed Processing Symposium.

Shen, L. (1998). *Logistics with competing users*. PhD thesis, University of Arizona, United States – Arizona.

Shen, W., Hao, Q., Yoon, H. J., and Norrie, D. H. (2006). Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics*, 20(4):415–431.

Soltani, R., Jolai, F., and Zandieh, M. (2010). Two robust meta-heuristics for scheduling multiple job classes on a single machine with multiple criteria. *Expert Systems with Applications*, 37(8):5951–5959.

Tan, Q., Chen, H. P., Du, B., and Li, X. L. (2011). Two-agent scheduling on a single batch processing machine with non-identical job sizes. 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce, AIMSEC 2011, pages 7431–7435.

T'kindt, V. and Billaut, J. C. (2001). Multicriteria scheduling problems: a survey. *Operations Research*, 35(2):143–163.

T'kindt, V. and Billaut, J. C. (2002). *Multicriteria scheduling: Theory, models and algorithms*. Springer, Berlin (Germany), second edition.

Unal, A. T., Uzsoy, R., and Kiran, A. S. (1997). Rescheduling on a single machine with part-type dependent setup times and deadlines. *Annals of Operations Research*, 70:93–113.

Wan, G., Vakati, S. R., Leung, J. Y. T., and Pinedo, M. L. (2010). Scheduling two agents with controllable processing times. *European Journal of Operational Research*, 205(3):528–539.

Warburton, A. (1987). Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1):70.

Wu, C. C., Huang, S. K., and Lee, W. C. (2011a). Two-agent scheduling with learning consideration. *Computers and Industrial Engineering*, 61(4):1324–1335.

Wu, W. H., Cheng, S. R., Wu, C. C., and Yin, Y. (2011b). Ant colony algorithms for a two-agent scheduling with sum-of processing times-based learning and deteriorating considerations. *Journal of Intelligent Manufacturing*, In press.

Yang, B. (2007). Single machine rescheduling with new jobs arrivals and processing time compression. *The International Journal of Advanced Manufacturing Technology*, 34(3-4):378–384.

Yazdani Sabouni, M. T. and Jolai, F. (2010). Optimal methods for batch processing problem with makespan and maximum lateness objectives. *Applied Mathematical Modelling*, 34(2):314–324.

Yin, Y., Cheng, S. R., and Wu, C. C. (2012a). Scheduling problems with two agents and a linear non-increasing deterioration to minimize earliness penalties. *Information Sciences*, 189:282–292.

Yin, Y., Wu, W. H., Cheng, S. R., and Wu, C. C. (2012b). An investigation on a two-agent single-machine scheduling problem with unequal release dates. *Computers and Operations Research*, In press.

Yuan, J. and Mu, Y. (2007). Rescheduling with release dates to minimize makespan under a limit on the maximum sequence disruption. *European Journal of Operational Research*, 182(2):936–944.

Yuan, J., Mu, Y., Lu, L., and Li, W. (2007). Rescheduling with release dates to minimize total sequence disruption under a limit on the makespan. *Asia - Pacific Journal of Operational Research*, 24(6):789–796.

Yuan, J. J., Shang, W. P., and Feng, Q. (2005). A note on the scheduling with two families of jobs. *Journal of Scheduling*, 8(6):537–542.

Zhao, C. and Tang, H. (2010). Rescheduling problems with deteriorating jobs under disruptions. *Applied Mathematical Modelling*, 34(1):238–243.